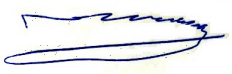
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Факультет електроніки
(повна назва інституту/факультету)

Кафедра акустичних та мультимедійних електронних систем
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри

Сергій Найда
(ініціали, прізвище)

“_1_”____червня____2020 р.

Дипломна робота

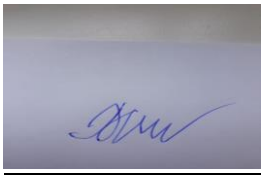
на здобуття ступеня бакалавра

зі спеціальності (спеціалізації) 171 Електроніка (Електронні та інформаційні системи і технології телебачення, кінематографії та звукотехніки)

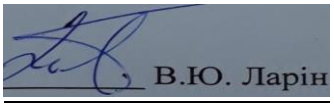
на тему: Застосування методу Віолі-Джонса для розпізнавання тривимірних об’єктів

Виконав: студент IV курсу, групи ___ДВ-61 _____
(шифр групи)

Бембель Олексій Сергійович _____ _____
(прізвище, ім’я, по батькові) (підпис)



Керівник доцент, к.т.н. Співак В.М _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)



Рецензент завідувач кафедри НАУ, д.т.н., професор _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет електроніки
(повна назва)

Кафедра акустичних та мультимедійних електронних систем
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність (спеціалізація) 171 – Електроніка (Електронні та інформаційні системи і технології телебачення, кінематографії та звукотехніки)
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Сергій Найда
(ініціали, прізвище)
«25» березня 2020 р.

ЗАВДАННЯ
на дипломну роботу студенту
Бембелю Олексію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: Застосування методу Віолі-Джонса для розпізнавання тривимірних об'єктів

керівник роботи Співак Віктор Михайлович, доцент, к.т.н. ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «25» березня 2020 р. №1196-с

2. Строк подання студентом роботи 1 червня 2020р.

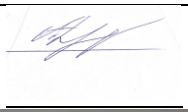
3. Вихідні дані до проекту (роботи)

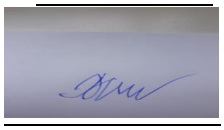
4. Зміст (дипломної роботи) пояснювальної записки (перелік завдань, які потрібно розробити) Алгоритми й методи виявлення обличчя; Методи розпізнавання обличчя, розробити систему, що дозволяє розпізнавати обличчя у відеопотоках у реальному часі з використанням методу Віоли-Джонса й локальних бінарних шаблонів

5. Перелік графічного (ілюстративного) матеріалу (із зазначенням обов’язкових креслеників, плакатів, презентацій тощо) презентація (9 слайдів).

6. Календарний план

з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Написання першого розділу: «Аналітичний огляд сучасних методів виявлення й розпізнавання обличчя у відеопотоках»	13.04.2020-17.05.2020	Виконано
2	Написання другого розділу: «Аналітичне дослідження методу віоли-джонса стосовно до розв'язку завдання розпізнавання обличчя у відеопотоках»	17.05.2020-20.05.2020	Виконано
3	Написання третього розділу: «Експериментальні дослідження системи розпізнавання обличчя людини»	20.05.2020-30.05.2020	Виконано
4	Підготовка матеріалів до друку та оформлення пояснювальної записки	30.06.2020-31.06.2020	Виконано
5	Підготовка та оформлення презентації для доповіді	31.06.2020-1.06.2020	

Студент  Бембель О.С.

Керівник роботи  Співак В.М.

РЕФЕРАТ

Дипломна робота: 115 с., 34 рисунки, 4 таблиці, 12 формул, 29 джерел.

Метою роботи є дослідження та удосконалення системи розпізнавання обличчя людини

Останнім часом широке поширення одержує відео аналітика – технологія використання методів комп'ютерного зору для автоматизованого збору різної інформації на основі послідовності кадрів, одержуваних з відеокамер у реальному часі або з відеозаписів.

На сьогоднішній день, така технологія може застосовуватися у відеоспостереженні на різного роду високотехнологічних проведеннях, системах безпеки, торгівлі, транспорті.

По оцінках компанії Marketsand Markets [1] у найближчі роки ринок відео аналітики продовжить активно рости, а до 2020 року складе 3971 мільйон доларів. Даний напрямок активний розвивається, у тому числі, і на російському ринку. Прикладом можуть служити технічні рішення, уже зараз надавані компаніями Синезис, Элвис-НеоТек. Одним із завдань, розв'язуваних відео аналітикою, є розпізнавання обличчя у відеопотоках. Розв'язок даного завдання в першу чергу має безпосереднє застосування в системах контролю доступу й ідентифікації обличчя.

Однак, на сьогоднішній день, набір існуючих методів і засобів для розв'язку завдань, пов'язаних з розпізнаванням обличчя у відеопотоках, з урахуванням складних умов відеоспостереження, нестабільністю умов навколишнього об'єкт простору, а також обмеженістю часу на обробку зображення, рухливістю об'єкта спостереження й іншими факторами, що ускладнюють роботу оптико-електронних систем відеоспостереження, усе ще далекі від досконалості.

У зв'язку із цим, завдання розробки й удосконалення системи розпізнавання обличчя людини є актуальною. Дана робота присвячена розробці

системи, що вирішує це завдання з використанням методу Віоли-Джонса й локальних бінарних шаблонів.

SUMMARY

The aim of the work is to study and improve the system of face recognition

Recently, video analytics has become widespread - the technology of using computer vision techniques to automatically collect various information based on a sequence of frames obtained from real-time video cameras or video recordings.

Today, this technology can be used in video surveillance on various high-tech wires, security systems, trade, transport.

According to Marketsand Markets [1], the video analytics market will continue to grow rapidly in the coming years, reaching $ 3,971 million by 2020. This area is actively developing, including in the Russian market. An example is the technical solutions already provided by Synesis, Elvis-NeoTech. One of the tasks solved by video analytics is face recognition in video streams. The solution of this problem first of all has direct application in systems of control of access and identification of the person.

However, to date, a set of existing methods and tools for solving problems related to face recognition in video streams, taking into account the complex conditions of video surveillance, the instability of the surrounding space object, as well as limited time for image processing, mobility Observation and other factors that complicate the operation of optoelectronic video surveillance systems are still far from perfect.

In this regard, the task of developing and improving the system of human face recognition is urgent. This work is devoted to the development of a system that solves this problem using the Viola-Jones method and local binary templates.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

1. Adaboost (Adaptive Boosting) – алгоритм посилення класифікаторів класифікаторів і розміру навчальної вибірки.

2. *Snow (Sparse Network of Winnows)* – алгоритм виявлення обличчя, що представляє собою двошарову мережу.

3. LBR - локальне бінарне перетворення.

ВСТУП

Актуальність роботи.

Розроблено систему розпізнавання обличчя в відеопотоках на основі методу Віоли-Джонса і локальних бінарних шаблонів. Для виявлення обличчя в кадрах відеопотоку був використаний метод Віоли-Джонса. Класифікація виявлених обличчя проводилася методом найближчого сусіда з використанням центрально-симетричних локальних бінарних шаблонів.

Тестування розробленої системи показало результати в приблизно 90% вірних розпізнаваних обличчя при обробці кадрів відеопотоку з вебкамери в реальному часі.

Проведено дослідження ефективності різних модифікацій локальних бінарних шаблонів стосовно до задачі розпізнавання обличчя в реальному часі. Було встановлено, що центрально-симетричні локальні бінарні шаблони практично не поступаються іншим варіаціям локальних бінарних шаблонів в якості ознак розпізнавання і при цьому мають набагато більш високу швидкість роботи. На підставі дослідження зроблено висновок, що гістограми центрально-симетричних локальних бінарних шаблонів є ефективною ознакою для класифікації обличчя в реальному часі.

Розроблено система може використовуватися при рішеннях різних завдань відеоаналітики, і в першу чергу, має безпосереднє застосування в системах контролю доступу та ідентифікації особистості. При цьому невисокі системні вимоги роблять можливим застосування розробки на системах з низькою продуктивністю.

Основними напрямками подальшого розвитку розробленого методу можна назвати поліпшення роботи класифікатора обличчя. Для цього доцільно використовувати більш досконалі алгоритми класифікації, ніж використаний метод найближчого сусіда, наприклад, такі методи як Random Forests і метод опорних векторів.

Мета роботи – розробити систему, що дозволяє розпізнавати обличчя у відеопотоках у реальному часі з використанням методу Віоли-Джонса й локальних бінарних шаблонів.

Для вирішення поставленої **мети** у роботі були поставлені та вирішені наступні задачі:

1. Виконати огляд існуючих методів і алгоритмів розпізнавання обличчя у відеопотоках, розпізнавання обличчя та реалізації з використанням сучасних електронно-оптичних систем;
2. Виконати аналітичні дослідження методу Віоли-Джонса стосовно до розв'язку завдання розпізнавання обличчя у відеопотоках.
3. Проаналізувати можливості використання локального бінарного перетворення в завданнях розпізнавання оптичних об'єктів
4. Розробити систему розпізнавання обличчя у відеопотоках і здійснити тестування роботи класифікатора обличчя.

Предмет дослідження. Оптико-електронна система розпізнавання обличчя у відеопотоках.

Об'єкт дослідження. Алгоритм роботи оптико-електронної системи розпізнавання обличчя у відеопотоці.

Методи дослідження базуються на теорії нечіткої логіки, математичної статистиці, імітаційного комп'ютерного моделюванні. Крім того, також застосовувалися вимір, опис, порівняння й аналіз, синтез інформації, використовувались методи створення нового програмного забезпечення, схемотехніки.

Наукова новизна досліджень полягає в тому, що: доведене, що гістограми центрально-симетричних локальних бінарних шаблонів є ефективним ознакою для класифікації обличчя у реальному часі.

Практична значимість дослідження полягає в тому, що розроблена система може використовуватися при рішеннях різних завдань відеоаналітики, і, у першу чергу, має безпосереднє застосування в системах контролю доступу та

ідентифікації обличчя. При цьому невисокі системні вимоги роблять можливим застосування розробки на системах з низькою продуктивністю.

Особистий внесок: у магістерській роботі розроблено систему розпізнавання обличчя у відеопотоках на основі використання методу Віоли-Джонса і локальних бінарних шаблонів. Для виявлення обличчя у кадрах відеопотоку був використаний метод локальних бінарних шаблонів. Класифікація виявлених обличчя проводилася методом найближчого сусіда з використанням центрально- симетричних локальних бінарних шаблонів. Тестування розробленої системи показало результати в приблизно 90 % вірних розпізнавань обличчя при обробці кадрів відеопотоку з вебкамери в реальному часі.

1. АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ МЕТОДІВ ВИЯВЛЕННЯ Й РОЗПІЗНАВАННЯ ОБЛИЧЧЯ У ВІДЕОПОТОКАХ

1.1 Алгоритми й методи виявлення обличчя

Процес (алгоритм) розпізнавання обличчя на кадрах відеопотоку умовно можна розділити на два етапи.

1-й – виявлення обличчя у кадрі,

2-й – безпосереднє розпізнавання знайдених обличчя. Із усього різноманіття існуючих алгоритмів виявлення обличчя можна виділити трохи найбільш актуальних методів, що й заслуговують уваги, [2]. Розглянемо особливості, гідності й недоліки кожного з них.

Метод Віоли-Джонса [3] був запропонований Полом Віолою й Майклом Джонсом в 2001 році й став першим методом, що демонструють високі результати при обробці зображень у реальному часі. В алгоритмі використовується набір ознак, близьких до ознак Хаара разом з варіацією алгоритму Adaboost.

Переваги:

- метод Віоли-Джонса є самим популярним і широко розповсюдженим методом виявлення обличчя;

- має високу швидкість виявлення за рахунок використання каскадного класифікатора, яка порівнянна з точністю виявлення в набагато більш повільних алгоритмів;

Недоліки:

- потрібно більша навчальна вибірка й великий час навчання;

- обмеження на положення обличчя при виявленні.

Adaboost (Adaptive Boosting) – алгоритм посилення класифікаторів шляхом об'єднання їх у комітет. Запропонований Йоавом Фройндом і Робертом Шапіре в 1999 році [7]. Може використовуватися в комбінації з декількома алгоритмами

класифікації для поліпшення їх ефективності. Даний алгоритм є адаптивним у тому розумінні, що кожний наступний комітет класифікаторів будується по об'єктах, невірно класифікованих попередніми комітетами. Найчастіше використовується в комбінації з іншими алгоритмами класифікації для їхнього посилення (наприклад, як було описано раніше, у методі Віоли-Джонса).

Переваги:

- підбудовується під проблемні елементи навчальної вибірки;

- теоретично метод досягає нульової помилки навчання за кінцеве число ітерацій;

- висока швидкість роботи й простота реалізації.

Недоліки:

- метод чутливий до шумів і викидам даних;

- час навчання, прямо залежать від кількості класифікаторів і розміру навчальної вибірки.

Snow (Sparse Network of Winnows) – алгоритм виявлення обличчя, що представляє собою двошарову мережу, вхідну верству якої складається з вузлів, кожний з яких відповідає деякій характеристиці вхідного зображення, вихідний же полягає всього із двох вузлів, кожний з яких відповідає розпізнаваним класам зображень [5]. У якості ознак у даному алгоритмі використовуються SMQT (Successive Mean Quantization Transform) ознаки. Таке перетворення дозволяє витягти з локальної області зображення корисну складову, не залежить від освітленості. Воно укладається у квантуванні області зображення з порогом квантування рівним середньому значенню пікселів, що входять у цю область.[6]

Переваги:

- використовувані ознаки нечутливі до змін висвітлення в локальних областях зображень;

- має високу швидкість роботи за рахунок просівання компонент вектора ознак і високою точністю виявлення.

Недоліком є те, що даний алгоритм чутливий до шумів і викидам даних.

Нейромережеві методи містять у собі цілий клас алгоритмів. Його основа – це послідовне перетворення сигналу паралельно працюючими функціональними елементами, нейронами. Сутність процесу навчання таких мереж зводиться до зменшення середньоквадратичної помилки. Системи виявлення об'єктів на зображеннях, засновані на нейронних мережах, використовують ієрархічну структуру. Спочатку вектор ознак обробляється грубою мережею з високим рівнем помилок другого роду, далі, якщо вектор не був класифікований як не об'єкт, розв'язок коректується більш точної й більш повільною мережею [7].

Перевагою є висока точність виявлення при правильному настроюванні параметрів мережі.

Недоліки

- чутливість до шуму;

- необхідність у ретельному й кропіткому настроюванні параметрів нейронної мережі для одержання гарних результатів;

- схильність до перенавчання;

- висока обчислювальна складність, і, як наслідок, швидкість роботи, недостатня для обробки в реальному часі.

Support vector machine – метод опорних векторів. Суть методу полягає в знаходженні гіперплощини, що розділяє два класи. При цьому із усіх можливих гіперплощин, що розділяють два класи, необхідно вибрати таку гіперплощину, відстань до якої від кожного класу максимально. Ця гіперплощина називається оптимальною поділяючою гіперплощиною, а відповідний їй лінійний класифікатор називається оптимально поділяючим класифікатором [8].

Переваги:

- дуже висока стабільність до перенавчання;

- можливість зменшення чутливості до шуму за рахунок зниження точності. Пошук оптимального співвідношення даних параметрів вимагає точного настроювання;

- висока швидкість роботи з порівняння з нейронними мережами.

Недоліки. Точність роботи даного методу уступає таким методам як Adaboost і Snow

Порівняння точності виявлень і розміру помилки другого при використанні описаних методів презентовано в таблиці 1.1.[9]

Таблиця 1.1 – Порівняння ефективності методів виявлення обличчя

Метод	Відсоток вірних виявлень	Помилка другого роду
Нейронні мережі	~92%	~1.3%
Метод опорних векторів	~72%	~0.6%
Snow	~94%	~0.12%
Adaboost	~94%	~0.00001%

Як видне із представлених даних, найкращим по показниках відсотка вірних виявлень і помилки другого роду є алгоритм Adaboost. Тому при створенні системи розпізнавання обличчя у відеопотоках будемо використовувати метод Віоли-Джонса, заснований на даному алгоритмі.

Крім переваг у точності виявлення, даний метод має високу швидкість роботи, що робить його найбільш підходящим для виявлення обличчя у реальному часі.

1.2 Методи розпізнавання обличчя

1.2.1 Методи, засновані на визначенні яскравості пікселів на зображенні

Методи розпізнавання обличчя можна умовно розділити на 2 підгрупи.

1-а – це методи, засновані на значеннях пікселів,

2-а - методи, засновані на характерних крапках [10]. Розглянемо їх більш докладно.

Назва даної групи методів має на увазі, що для розпізнавання виявлених обличь використовується тільки колір або яскравість пікселів на зображеннях. Найпростішим подібним методом є порівняння, у якому заходом схожості є відстань між векторами яскравості пікселів зображень. Однак даний метод абсолютно нестійкий до змін висвітлення, положення обличчя, масштабуванню. Більше того, такий підхід має високу обчислювальну складність і зовсім не придатний для розпізнавання в реальному часі. Тому, дуже часто використовуються методи, що використовують перехід векторного опису зображень у простори з меншою розмірністю, у яких порівняння набагато ефективніше [10].

Eigenfaces – алгоритм, запропонований в 1991 році Метью Тѐрком і Алексом Пентландом [11], що й одержав широку популярність у якості першого успішного методу розпізнавання обличчя. Основною ідеєю алгоритму є застосування методу головних компонентів для знаходження векторів, що щонайкраще описують зображення обличчя. Використовуючи цей метод можна виявити різні мінливості в навчальній вибірці зображень обличчя і описати цю мінливість у базисі декількох ортогональних векторів, які називаються власними (eigenface).

Отриманий один раз на навчальній вибірці зображень обличчя набір власних векторів використовується для кодування всіх інших зображень обличчя, які представляються зваженою комбінацією цих власних векторів. Використовуючи обмежену кількість власних векторів можна одержати стислу апроксимацію вхідному зображенню обличчя, яку потім можна зберігати в базі даних у вигляді вектору коефіцієнтів, що служить одночасно ключем пошуку в базі даних обличчя [11].

1.2.2 Методи, засновані на характерних крапках

Дана група методів, на відміну від попередньої, використовує характерні крапки і їх координати на зображенні, а не оцінює яскравості пікселів. Такими

характерними крапками можуть бути, наприклад, центри очей, положення носа, лінія брів, рота і т. д. [10]. До даного класу методів ставляться активні моделі зовнішнього вигляду й активні моделі форми.

Активні моделі зовнішнього вигляду (Active Appearance Models, AAM) — це статистичні моделі зображень, які шляхом різного роду деформацій можуть бути підігнані під реальне зображення. Даний тип моделей у двовимірному варіанті був запропонований Тімом Кутсом і Крисом Тейлором в 1998 році [15]. Активна модель зовнішнього вигляду містить два типи параметрів: параметри, пов'язані з формою (параметри форми), і параметри, зв'язані зі статистичною моделлю пікселів зображення або текстурою (параметри зовнішнього вигляду). Перед використанням модель повинна бути навчена на безлічі заздалегідь розмічених зображень. Розмітка зображень проводиться вручну.

Активні моделі форми (Active Shape Models, ASM) враховують статистичні зв'язки у взаємному розташуванні антропометричних крапок. На кожному зображенні вибірки експерт розмічає розташування антропометричних крапок. Для того щоб привести координати на всіх зображеннях до єдиної системи звичайно виконується так званий узагальнений «прокрустів» аналіз, у результаті якого всі крапки приводяться по одному масштабу й центруються. Далі для всього набору образів обчислюється середня форма й матриця коваріації. На основі матриці обчислюються власні вектори, які потім сортуються в порядку убування відповідних їм власних значень. Локалізації ASM моделі на новому зображенні, що не входить у навчальну вибірку, здійснюється в процесі розв'язку оптимізаційного завдання [16].

Однак варто відзначити, що подібні моделі споконвічно призначені не для розпізнавання, а для точної локалізації характерних крапок на зображеннях обличчя. Їхня локалізація дозволить виконати процедуру вирівнювання обличчя вибірки й приведення їх до одній системі координат для більш точного розпізнавання іншими методами. Звичайно для цих цілей використовується невелика кількість крапок, що дозволяє прискорити продуктивність. Для завдань

розпізнавання, навпаки, потрібне велика кількість характерних крапок, що побільшає точність класифікації й знизить швидкість роботи системи [17].

1.2.3 Вейвлет- перетворення

Вейвлет-перетворення широко використовується для аналізу нестаціонарних процесів. Воно показало свою ефективність для розв'язку широкого класу завдань, пов'язаних з обробкою зображення. Коефіцієнти вейвлет - перетворення містять інформацію про аналізований процес і використовуваному вейвлеті. Тому вибір, що аналізує вейвлету визначається тем, яку інформацію необхідно витягти із процесу. Кожний вейвлет має характерні риси в тимчасовий і частотної областях, тому іноді за допомогою різних вейвлетів можна повніше ви виявити й підкреслити ті або інші властивості аналізованого процесу. У роботах [7, 8] представлені розкладання зображення й витяг його ознак для класифікації зображень літаків на основі застосування Вейвлет- перетворення Хаара й багатошарової нейронної мережі. У даній роботі використовуються Вейвлет- перетворення Хаара й Добеши для визначення ознак зображення обличчя. Приклади застосування Вейвлет- перетворення Хаара й Вейвлет- перетворення Добеши для витягу ознак зображення обличчя представлені на рисунку 1.1.

Рис. 1.1 – Приклад витягу ознак обличчя: а) вихідне зображення обличчя; б) результат після застосування Вейвлет- перетворення Хаара

1.2.4 Метод головних компонентів

Метод головних компонентів добре зарекомендував себе в практичних додатках. Однак, у тих випадках, коли на зображенні обличчя присутні значні зміни в освітленості або вираженні обличчя, ефективність методу значно падає.

До гідності даного методу можна віднести простоту реалізації, придатність для розпізнавання в реальному часі й можливість компактно зберігати більші обсяги даних [10].

Основним недоліком же є висока чутливість до зміни висвітлення, розміру й поворотам, і, як результат, необхідність строгого збереження вихідних умов зйомки. Дана проблема викликана тим, що найбільш важливі власні вектори більшою мірою описують особливості висвітлення, ніж характеристики обличчя, оскільки споконвічно метод головних компонентів вибирає підпростір з метою апроксимації даних, а не їхні класифікації [10].

Fisherfaces – алгоритм, у якому на відміну від методу eigenfaces використовується лінійний дискримінантний аналіз, а саме лінійний дискримінант Фішера. Дія алгоритму заснована на пошуку проекції даних, при якій класи зображень обличчя максимально роздільні. При використанні ж методу головних компонентів проводиться максимізація розкиду даних по всій базі обличчя. Дана відмінність дозволяє розв'язати проблему високої чутливості до змін висвітлення. [12]. Локальні бінарні шаблони(надалі LBP – Local Binary Pattern) – простий і ефективний оператор перетворення зображень, уперше запропонований в 1996 році для класифікації текстур [13]. Однак, пізніше знайшов застосування й для розпізнавання обличчя [14].

Даний оператор використовує значення яскравості околиці кожного пікселя зображення й за допомогою функції привласнює кожному пікселю значення, що описує його околиця. Далі отримане зображення розділяється на під області, для кожної з яких розраховується гістограма. Гістограми рівняються за допомогою методів машинного навчання. У класичному варіанті використовується метод найближчого сусіда [14].

Переваги даного методу укладаються в простоті реалізації й високої швидкості роботи, яку можна побільшати, використовуючи різні модифікації алгоритму. При цьому алгоритм показує високі результати при розпізнаванні обличчя і стійкий до монотонних змін висвітлення. Усе це робить його ідеально підходящим для розпізнавання обличчя у системах обробки в реальному часі. Метод головних компонентів (*Principal Component Analysis*, PCA) – один з найпоширеніших методів для зменшення розмірності даних, втрати найменшої кількості інформації. Він укладається в лінійному ортогональному перетворенні вхідного вектору P розмірності N у вихідний вектор Q розмірності M, $M<N$. Компоненти вектору Q є не корельованими, і загальна дисперсія після перетворення залишається незмінної.

Головна ідея МГК полягає у виставі зображень обличчя людей у вигляді набору головних компонентів зображень, називаних «власні обличчя» (*Eigenfaces*). Власні обличчя мають корисну властивість, що укладається в тому, що зображення, що відповідає кожному такому вектору має обличчя-подібну форму, рисунок. 1.2.

Обчислення головних компонентів зводиться до обчислення власних векторів і власних значень матриці, яка розраховується із зображення. Сума головних компонентів, помножених на відповідні власні вектори, є реконструкцією зображення, яке показано рисунку 1.2.

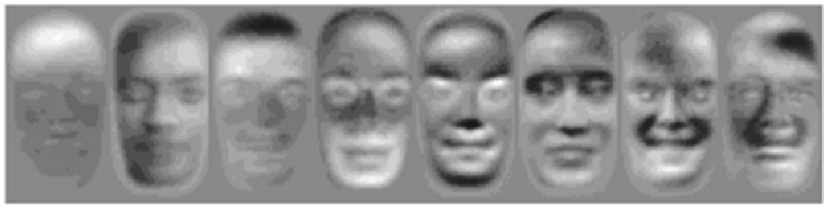


Рис. 1.2 – Приклад зображень власних векторів (власні обличчя)

Рис. 1.3 – Приклад зображень обличчя: а) вирівняне зображення; б) реконструкція по головним компонентам; в) Jpeg - реконструкція (530 байт)

Для кожного зображення обличчя обчислюються його головні компоненти. Звичайно береться від 5 до 200 головних компонентів. Інші компоненти кодують дрібні відмінності між обличчями й шумами. Процес розпізнавання укладається в порівнянні головних компонентів невідомого зображення з компонентами всіх відомих зображень. При цьому передбачається, що зображення облич, що відповідають одній людині, згруповані в кластери у власному просторі. З бази даних вибираються зображення кандидати, що мають найменшу відстань від вхідного (не відомого) зображення [9].

Висновки по розділу 1

1. Виконаний аналіз сучасних методів виявлення обличчя, розглянуті їхні гідності й недоліки.
2. Зроблений висновок про те, що найбільш підходящим методом для обробки зображень у реальному масштабі часу є метод з використанням локальних бінарних шаблонів
3. Обґрунтований вибір методу Віоли-Джонса для реалізації алгоритму в розроблювальній системі виявлення обличчя на основі локальних бінарних шаблонів як алгоритму розпізнавання.

2. АНАЛІТИЧНЕ ДОСЛІДЖЕННЯ МЕТОДУ ВІОЛИ-ДЖОНСА СТОСОВНО ДО РОЗВ'ЯЗКУ ЗАВДАННЯ РОЗПІЗНАВАННЯ ОБЛИЧЧЯ У ВІДЕОПОТОКАХ

2.1. Постановка завдання досліджень

Представимо структуру системи для розпізнавання образів. Така система повинна провадити обробку й аналіз зображень, що надходять від оптико-електронної системи.

Враховуючи характер розв'язуваних завдань, необхідно врахувати наступні обмеження й особливості:

1) Інформація про характеристики обличчя й спостережуваних об'єктів найчастіше містить неточні розміри об'єктів або зовсім відсутня;

2) У зв'язку з обмеженістю часу на прийняття рішення і наявністю замкненого контуру керування рухом обробка, аналіз і розпізнавання повинні виконуватися в реальному часі;

3) Робота системи керування, спостереження й обробки повинна проводитися при мінімальній участі людини або в автономному режимі.

У складних умовах, поліпшити якість сигналу й результату спостереження можна за допомогою алгоритмів оцінювання параметрів геометричних перетворень зображень і методів просторово-тимчасової фільтрації. Серед них можна виділити чотири основні підходи.

1. Для виміру положення нерухливих об'єктів, що й рухаються, спостережуваних на однорідному й неоднорідному тлі використовуються методи на основі порівняння з еталоном.

2. Використовуючи інформацію про статистичні властивості об'єкту й фону можна виділяти нерухливі об'єкти, що рухаються й, спостережувані на порівняно однорідному фоні. Такі методи називаються методами статистичної сегментації.

3. При виявленні об'єктів на фоні ясного або хмарного неба максимальну ефективність показують методи виділення об'єктів за допомогою просторової фільтрації. Такий підхід використовує операції лінійної й нелінійної просторової фільтрації зображень.

4. Виділення динамічних змін на основі фіксування змін, що відбуваються із часом у спостережуваній групі зображень. Такі методи застосовуються при розв'язку завдання виділення об'єктів, що рухаються.

На рисунку 2.1 представлена структура системи виявлення об'єктів, яка включає перераховані вище концепції.

Рис. 2.1 – Структура оптико-електронної системи виявлення й розпізнавання обличчя

2.1.1 Реалізація алгоритму розпізнавання об'єктів зовнішнього середовища

У процесі попередньої підготовки над відео зображенням звичайно проводяться наступні операції:

– перетворення кольорового 4-канального зображення в чорно-біле одноканальне;

- цифрова корекція аберації об'єктива телекамери;
- фільтрація відео зображення.

Вирішуватися ці завдання повинні в реальному часі при наявності обмежень на обчислювальні ресурси.

Захоплення відеозображення.

Перетворення кольорового зображення в чорно – біле. При обмежених обчислювальних ресурсах важливо вибрати припустимий розмір відеозображення в пікселях, а також для зберігання в пам'яті організувати структуру даних, яка б забезпечувала швидкий доступ до зображення й зручну роботи з ним. Для цього в бібліотеці Opencv [https://opencv.org/] є спеціальний тип даних – структура Iplimage. По суті це матриця, інтерпретована як зображення.

Для розв'язку завдань виявлення перешкод і розпізнавання образів необхідна наявність неспотвореного зображення. Однак більшість об'єктивів телекамер мають ефект, що спотворює (аберацією). Об'єктиви, у яких повністю усунута аберація, мають високу вартість і більшими масо габаритними характеристиками. Для усунення аберації об'єктива телекамери пропонується використання алгоритму цифрової корекції викривлень на відеозображенні. Необхідною вимогою до цих алгоритмів є досягнення максимальної швидкодії. Opencv надає нам готовий до використання алгоритм виправлення викривлень (undistortion), який ухвалює сирі зображення й коефіцієнти викривлення з *cvcalibratecamera2* і створює виправлене зображення. При побудові інтелектуального мобільного робота з високим ступенем інтелектуальності необхідна така система розпізнавання образів, яка дозволила б не тільки розпізнавати відомі їй образи, але й навчатися новим.

Розглянемо загальний алгоритм розпізнавання. Загальна послідовність дії при розпізнаванні виглядає так:

- попередня обробка зображення – згладжування, фільтрація перешкод, підвищення контрасту;
- бінарізація зображення й виділення контурів об'єктів;

– початкова фільтрація контурів по периметру, площі, коефіцієнті форми;

– приведення контурів до єдиної довжини, згладжування;

– перебір усіх знайдених контурів, пошук шаблону, максимально схожого на даний контур.

Бібліотека Opencv надає можливість розроблювачам легко детектувати контури зображення й маніпулювати ними. Для пошуку контурів пропонується використовувати функцію *cvfindcontours*. Функція *cvfindcontours* відшукує контури від монохромного зображення й повертає число знайдених контурів. Після того, як контури виявлені – їх можна вивести у вигляді зображення за допомогою функції *cvdrawcontours*. Для згладжування й одержання більш акуратних контурів можна скористатися функцією *cvapproxpoly*. У випадку руху камери на якому-небудь об'єкті, важливо визначити напрямок руху. У цьому випадку можна використовувати наступну функцію в Opencv – *phasecorrelate*. Функція використовується для визначення зрушень між двома зображеннями (масивами).

На рисунку 2.2. представлений результат роботи алгоритму перекладу зображення в екранні образи.

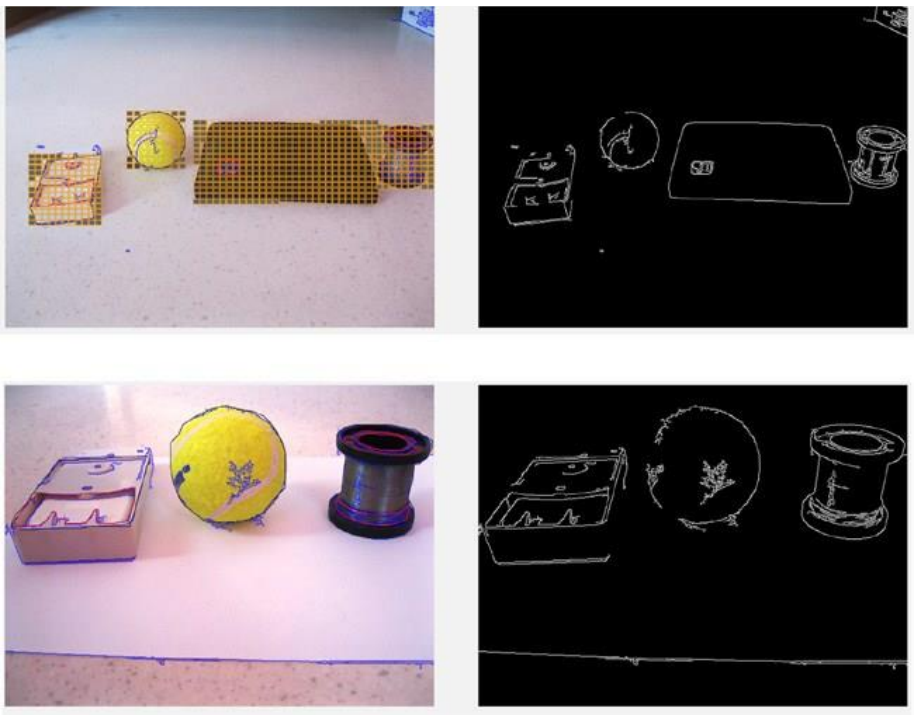


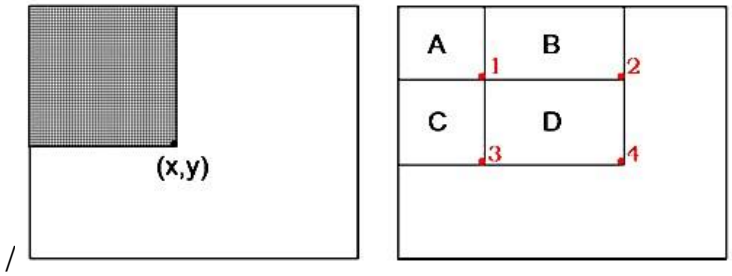
Рис. 2.2 – Результат роботи алгоритму перекладу зображення в екранні образи

2.2. Дослідження можливостей використання методу Віоли-Джонса для розв'язку задач виявлення обличчя

За даними роботи [12, 13], перевагами методу Віоли-Джонса для розв'язку завдань виявлення обличчя є його високі швидкість роботи й відсоток імовірності вірного детектування об'єкта на зображенні. Таким чином, даний метод можна застосовувати в багатьох сферах діяльності, наприклад, для виявлення розпізнавання номерних знаків на автомобілях (для подальшого їхнього аналізу) або виявлення обличчя на зображенні.

П. Віола й М. Джонс застосували метод інтегрального уявлення зображення. Вага кожного пікселя являє собою його яскравість. Інтегральне значення для кожного пікселя є сума всіх значень над ним і ліворуч від нього плюс його власна вага. Починаючи з лівого верхнього кута вправо й униз, усе зображення може бути інтегроване з декількома цілочисельними операціями.

Як показано на рисунку 2.3, a, після інтеграції значення кожного пікселя з координатами (x, y) містить суму всіх піксельних значень усередині прямокутної області, яка має один кут у лівій верхній частині зображення й іншої в положенні (x, y). Щоб знайти середнє піксельне значення в цьому прямокутнику, необхідно тільки розділити значення в прямокутнику (x, y) на площу прямокутника. Далі знайдемо сумарні значення для деяких інших прямокутників, які не мають жодного кута в лівій верхній частині зображення. Рисунок 2.3. b ілюструє даний розв'язок. Припустимо, що потрібно довідатися підсумовані значення в області D. І що це сума значень пікселів у комбінованому прямокутнику, A+B+З+D, мінус сума в прямокутниках A+B і A+C, плюс сума піксельних значень у прямокутнику A. Так, з інтегральним зображенням можна знайти суму піксельних значень для будь-якого прямокутника в первісному зображенні всього за допомогою трьох чисельних операцій.



А) Б)

Рис. 2.3 – Інтегральне зображення

Особливості, які використовували П. Віола й М. Джонс, базуються на каскадах ознак Хаару. Основною причиною, використання примітивів Хаару в основі методу Віоли- Джонса була спроба піти від пиксельного вистави зі збереженням швидкості обчислення ознаки. Така назва вони одержали через те, що їх основою є вейвлети Хаару. Каскади Хаару являють собою прямокутні області, які складені з декількох сусідніх прямокутних областей, відзначених як світла або темна. На рисунку 2.4 показані функції, що використовуються в Opencv (в оригінальному методі Віоли-Джонса похилих областей не було).

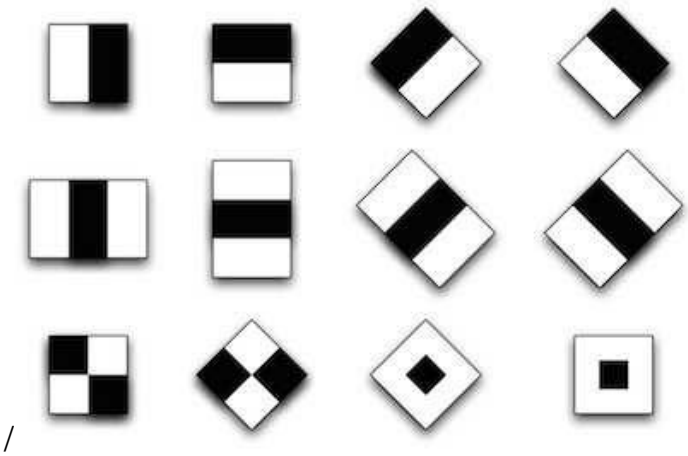


Рис. 2.4 – Приклади функцій Хаару, використовуваних в Opencv

Зі значень пари пікселів складно винести яку-небудь осмислену інформацію для класифікації, у той час як із двох ознак Хаару будується, наприклад, перший каскад системи по розпізнаванню обличчя, який має цілком осмислену інтерпретацію (рис. 2.5) [14]. Наявність функції Хаару визначається за допомогою вирахування середнього значення області темних пікселів із

середнього значення області світлих пікселів. Якщо різниця перевищує поріг (обумовлений у процесі навчання), то говорять, що функція є існуючою.

Ще однією важливою перевагою каскадів перед піксельним виставою є те, що складність обчислення ознаки, як і одержання значення пікселя, залишається O(1).

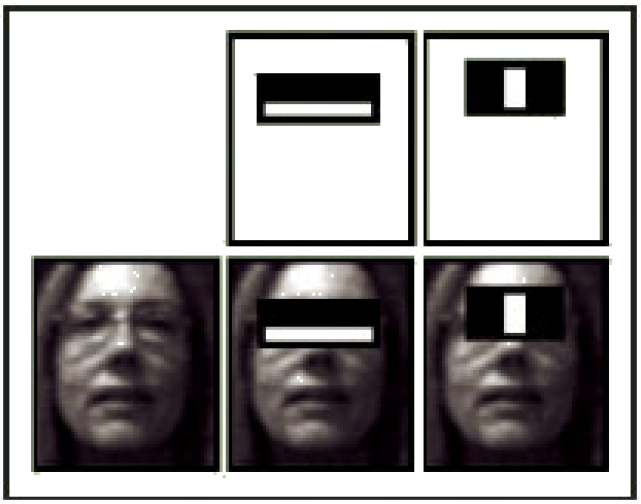


Рис. 2.5. – Перші дві Хаар - функції в оригінальному каскаді Віоли-Джонса

2.3. Метод машинного навчання Adaboost

Для вибору конкретних використовуваних функцій Хаару й установлення граничних рівнів, П. Віола й М. Джонс використовували метод машинного навчання Adaboost. Він комбінує багато «слабких» класифікаторів з метою створення одного «сильного». «Слабкий» тут означає такий класифікатор, який одержує правильну відповідь не набагато частіше, чим випадкове вгадування, що є недоліком. Однак маючи безліч таких слабких класифікаторів, кожний з яких «висунув» остаточна відповідь небагато у вірному напрямку, можна одержати серйозну комбіновану чинність для досягнення коректного розв'язку. Adaboost вибирає набір слабких класифікаторів для об'єднання й привласнює кожному з них своя вага. Ця зважена комбінація і є сильним класифікатором. П. Віола й М. Джонс об'єднали серії класифікаторів Adaboost як послідовність фільтрів, що особливо ефективно для класифікації областей зображення. Кожний фільтр є

окремим класифікатором Adaboost з досить малим числом слабких класифікаторів.

Каскадний класифікатор

Порядок фільтрів у каскаді ґрунтується на вагових значеннях, які привласнює Adaboost. Більш важкі зважені фільтри розташовуються на початку, для більш швидкого усунення (відкидання) областей зображення, що не містять даний об'єкт зображення («не особових» областей). На рисунку 2.5 показане накладення перших двох функцій з каскаду Віоли-Джонса на обличчя. Перша функція заснована на факті того, що область очей більш світла, чому область верхньої частини щік, а друга перевіряє наявність більш світлого перенісся між двома темними областями очей. Прийнятий поріг на кожному рівні встановлюється досить низьким, щоб пройти всі (або майже все) особові зразки в тренувальному наборі. Фільтри на кожному рівні можуть класифікувати тренувальні зображення, які пройшли всі попередні етапи. Якщо під час роботи якийсь із цих фільтрів не пропускає область зображення, то тоді область відразу ж класифікується як «не обличчя». Коли фільтр пропускає область зображення, вона переходить до наступного фільтра в послідовності. Області зображення, що пройшли через усі фільтри, класифікуються як «обличчя». П. Віола й М. Джонс назвали це фільтрацією ланцюга каскаду. Кожний наступний рівень ланцюга каскаду містить усе більше класифікаторів, наприклад каскад для розпізнавання обличчя в анфас на перших рівнях містить тільки один або два самі «сильні» класифікатори, а останні – до 200, але «слабких».

2.3.1 Дослідження можливостей практичного застосування методу Віоли-Джонса для розпізнавання обличчя

Алгоритм розпізнавання обличчя по методу Віоли-Джонса може застосовуватися в різних сферах для розв'язку найрізноманітніших завдань. Його найпоширеніша реалізація належить бібліотеці комп'ютерного зору Opencv. Класичним прикладом для демонстрації роботи методу Віоли-Джонса є

програма розпізнання обличчя у фас на світлині [3]. Діаграма класів подібної програми зображена на рисунку 2.6. Дана програма дозволяє визначити координати центрів обличчя, розташованих на оброблюваному зображенні. Її перевагою є можливість швидко розпізнавати об'єкти типу «обличчя» на зображенні, що дозволяє використовувати подібний алгоритм при аналізі даних, що надходять у реальному часі, як при обробці відео.

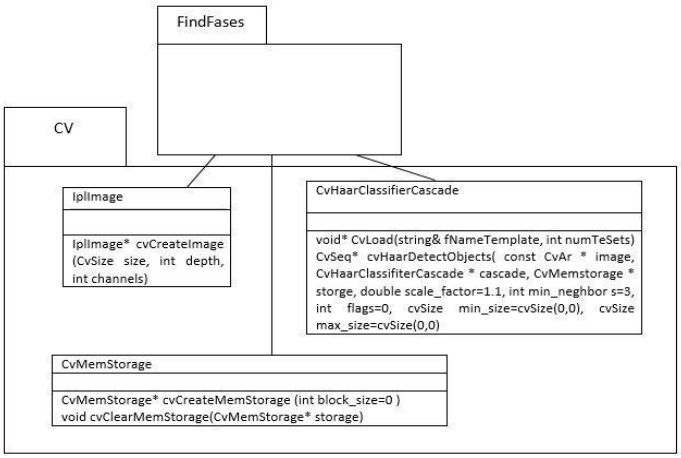


Рис. 2.6 – Діаграма класів програми, що реалізує метод Віоли-Джонса з використанням бібліотеки Open для мови С++

Перевагами такого методу є висока ймовірність вірного розпізнання об'єктів і можливість порівняно простої зміни типу розпізнаваного об'єкта при наявності класифікатора), наприклад, на номерні знаки. Недоліками ж є різке погіршення ймовірності виявлення об'єкта при посиленні шумів на зображенні, а також висока складність створення нових класифікаторів розпізнавання інших об'єктів, які будуть використані в процесі машинного навчання.

Обробка кадрів відеопотоку досліджуваною системою повинна включати два основні етапи. Перший етап - виявлення обличчя методом Віоли-Джонса. Другий етап – розпізнавання знайдених обличчя за допомогою гістограми локальних бінарних шаблонів і методу найближчого сусіда. Однак продуктивність даних алгоритмів суттєво залежить від таких факторів як

висвітлення, положення обличчя і т. д. Тому, доцільно відразу описати умови застосування розроблювальної системи, у яких може бути забезпечена її коректна робота:

- припустима тільки монотонна зміна висвітлення. Навчальна й тестова вибірка повинна зніматися в однакових умовах висвітлення;

- використовується фронтальне, або близькі до нього положення облич;

- необхідно нейтральне вираження обличчя у зображеннях;

- обличчя не повинні перекриватися іншими об'єктами.

Також доцільно відразу описати необхідний функціонал розроблювальної системи:

- обробка відеопотоку з підключеної до комп'ютера камери в реальному часі;

- можливість настроювання параметрів роботи використовуваних для виявлення й розпізнавання алгоритмів;

- висновок інформації про розпізнане обличчя, що включає захід приналежності до певного класу, графічне відображення гістограми й LBP вистава обличчя, що відслідковується.

- можливість навчання й додавання класів обличчя з використанням камери через інтерфейс додатка. Крім етапів виявлення й розпізнавання доцільно використовувати проміжні етапи обробки знайдених облич. Застосування фільтра Гауса після виявлення обличчя допоможе знизити вплив шумів при розпізнаванні.

Також має сенс застосувати значимих областей до локалізованим і перетвореним оператором LBP- зображенням обличчя, яке дозволить забрати вплив при розпізнаванні кутових областей зображення, що містять задній план.

У результаті загальний алгоритм розпізнавання повинен містити наступні кроки: виявлення обличчя у кадрові, обробка знайдених облич фільтром Гауса, застосування LBP- трансформації до знайдених обличчя з наступним застосуванням маски значимих областей, розрахунки гістограм знайдених обличчя, класифікація обличчя по гістограмам методом найближчого сусіда.

У підсумку буде отриманий список облич, що відслідковуються, з їхніми характеристиками й координатами прямокутних областей кадру, у яких вони перебувають. Узагальнена блок-схема алгоритму обробки кадрів розроблювальною системою представлена на рисунку 2.7.

Процедура навчання проводиться аналогічним образом. Знайдене в кадрі обличчя послідовно обробляється відповідно до описаного алгоритму, розраховані гістограми обличчя навчальної вибірки кожного класу зберігаються. Розпізнавання проводиться на основі пошуку мінімальної відстані між гістограмою вхідного зображення обличчя й гістограм, що зберігаються в електронній базі. [30]

Рис. 2.7 – Узагальнена блок-схема алгоритму обробки кадрів відеопотоку

2.4. Аналіз ефективності оптико-електронної системи розпізнавання об'єктів

На рисунку. 2.8 [15], наведена функціональна схема оптико-електронної системи, що розпізнає. Незважаючи на різноманітність систем, що розпізнають, по призначенню й використанню фізичних ознак об'єктів розпізнавання, у них можна виділити загальні для всіх систем блоки. Блоки чутливих пристроїв служать для перетворення оптичного випромінювання від об'єктів в електричний сигнал. Система може мати один або кілька чутливих пристроїв, кожний з яких сприймає променистий потік від об'єкту в певному спектральному діапазоні.

Сигнали на виході рецепторів пропорційні ефективним променистим потокам від цілей і фонів. Розв'язок про приналежність виявлених сигналів до певного класу об'єктів може здійснюватися двояким образом: тільки по амплітудах сигналів (тобто по ефективних променистих потоках) без залучення додаткової інформації й по тим же сигналам, але із залученням додаткової інформації, такий, як аналіз форми об'єкту, аналіз координат джерела випромінювання і їх похідних і т.п.

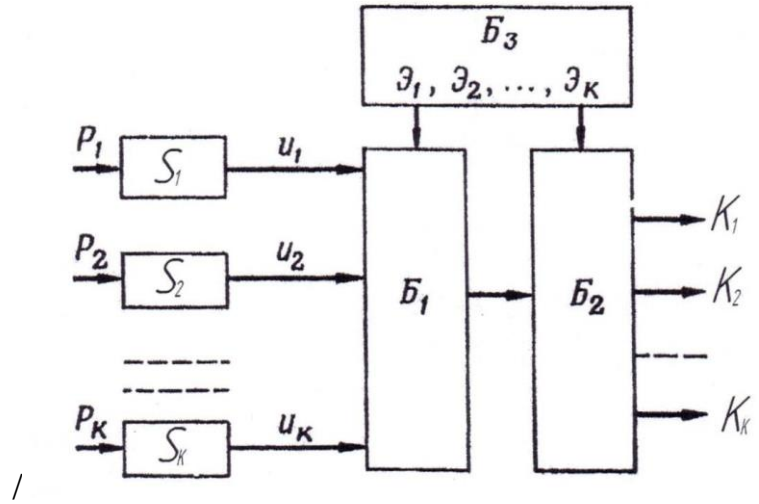


Рис. 2.8 – Функціональна схема оптико-електронного пристрою, що розпізнає

Обробка сигналів по первинних ознаках проводиться в логічному блоці $Б_1$ шляхом порівняння з еталонами, що зберігаються в блоці пам'яті $Б_3$, або в блоці $Б_1$ і блоці порівняння з побічними ознаками $Б_2$. На виході блоків $Б_1$ і $Б_2$ видається розв'язок про приналежність поточного значення сигналу до того або іншому класу об'єктів $(К_1, К_2, \ldots, К_к)$. Вирішальним правилом розпізнавання є мінімально-задана відмінність ознак об'єкту й еталону даного класу. Оптичні пристрої обробки інформації забезпечують більшу швидкість, продуктивність, паралелізм обчислень. Одним з багатообіцяючих напрямків побудови систем розпізнавання обличчя, є використання нейронних мереж, які забезпечують гнучкість, адаптивність зовнішнім умовам, висока якість.

Історично зложилося, що теорія розпізнавання образів розвилася по двом напрямкам: детермінованому й статистичному, хоча на практиці знаходить застосування комбінація цих напрямків. Детермінований підхід включає такі методи, як емпіричні, евристичні, які засновані на логіку, теорії графів, кластерному аналізі, топології й ін. Статистичні методи опираються на математичну статистику (теорія оцінок, Байєсове правило, послідовний аналіз, теорія ймовірностей, стохастична апроксимація й ін.).

У статистичній теорії ознаки об'єктів і еталонів у процесі розпізнавання можуть мінятися в часі. Ці зміни, як правило, носять випадковий характер і підкоряються нормальному закону розподілу. І для їхньої порівняльної оцінки необхідно розташовувати густинами розподілу і їх числовими показниками.

Відповідно до принципу дихотомії (спосіб логічного розподілу на два взаємовиключні поняття) будь-яка безліч об'єктів, які реєструє оптико-електронна система, можна розділити на істинні й хибні. Тоді апріорна інформація може бути представлена наступними показниками – щільність розподілу ознак розпізнавання для істинних і хибних об'єктів відповідно; П – вектор ознак розпізнавання (наприклад, дальність первинного виявлення, параметри відбитого сигналу й т.п.) – імовірності появи істинного й хибного об'єктів.

Дихотомічний розподіл привабливий своєю простотою, тобто розподіл проводиться по наявності або відсутності тієї або іншої ознаки на два поняття, які виключають одну («істинна», «хибна»). Недолік цього принципу – наявність фактору невизначеності, особливо в понятті «хибна». Наявність невизначеності обумовлена [3, 5, 6, 7]: недостатньою вивченістю ознак (фізичних явищ, які, як правило, мають стохастичну природу) тобто апріорної непоінформованості про ознаки об'єкта розпізнавання;по відношенню характеристик технічних пристроїв розпізнавання; нечіткістю постановки завдання дослідження й необхідністю пошуку оптимальних рішень по декільком критеріям.

Проблема обліку невизначених факторів є настільки значимою, що до її розв'язку, як правило, необхідно приділити особливої увагу на всіх етапах досліджень – від формулювання завдання до аналізу отриманих результатів і ухвалення рішення. У математичній моделі розпізнавання образів необхідно враховувати не тільки випадкові, але й реальні невизначеності, як по відношенню обраних ознак, так і по відношенню технічних характеристик коштів розпізнавання й умов функціонування.

В основі статистичної теорії розпізнавання лежить правило мінімізації середнього ризику, яке називається байесовим правилом або байесовським методом оцінки ймовірності різних гіпотез [1, 4, 5]. Якщо в якості середнього ризику розглядається ймовірність того, що істинний об'єкт прийнятий за неправильний або навпаки, то байесово правило можна сформулювати таким чином, якщо, те об'єкт істинний, а якщо ні, то – хибний: тут - конкретне значення ознаки розпізнавання. Відношення густин істинного при певному значенні вектору П називається відношенням правдоподібності й позначається. Для оцінки технічної подоби при зіставленні двох об'єктів інтерес представляє випадок. Об'єкт вважається істинним, якщо, а якщо ні, то – хибним. Це відповідає граничному принципу розпізнавання.

У загальному випадку, використовуючи формулу Байеса, можна визначити ймовірність приналежності об'єкта до класу істинних або хибних відповідно, якщо ознака розпізнавання прийняла значення.

$$P(\text{И}, \text{П}^*) = \frac{P_{\text{И}} f_{\text{И}}(\text{П}^*)}{P_{\text{И}} f_{\text{И}}(\text{П}^*) + P_{\text{Л}} f_{\text{Л}}(\text{П}^*)};$$
$$P(\text{Л}, \text{П}^*) = \frac{P_{\text{Л}} f_{\text{Л}}(\text{П}^*)}{P_{\text{И}} f_{\text{И}}(\text{П}^*) + P_{\text{Л}} f_{\text{Л}}(\text{П}^*)}. \quad (2.1)$$

Очевидно, що ці залежності характеризують конкретну реалізацію розпізнавання з урахуванням конкретних значень ознак і використовуються в алгоритмах систем розпізнавання.

Для аналізу технічної подоби на етапі проектування звичайно визначається середня ймовірність розпізнавання у всьому діапазоні ознак П. Якщо врахувати апріорні розподіли цих ознак для кожного класу об'єктів, можна одержати показники технічної подоби:

- імовірність прийняття дійсного об'єкта за істинний:

$$P_{\text{и/и}} = \int_{S_{\text{П}}} P(\text{И}, \text{П}) f_{\text{И}}(\text{П}) d\text{П}; \quad (2.2)$$

- імовірність прийняття дійсного об'єкта за хибний:

$$P_{\text{и/л}} = \int_{S_{\text{П}}} P(\text{Л}, \text{П}) f_{\text{И}}(\text{П}) d\text{П}; \quad (2.3)$$

- імовірність прийняття неправильного за хибний:

- імовірність прийняття неправильного за істинний:

$$P_{\text{л/л}} = \int_{S_{\text{П}}} P(\text{Л}, \text{П}) f_{\text{Л}}(\text{П}) d\text{П}; \quad (2.4)$$

Враховуючи співвідношення, можна визначати тільки ймовірності розпізнавання і. Тут $f_л(П)$– область можливої зміни ознак розпізнавання. Діапазон зміни цих показників при такий: при абсолютній подобі, тобто при (рисунок 2.9, а), маємо ситуацію повної невизначеності; при абсолютній

відмінності (істотна відмінність ознак) маємо ситуацію повної визначеності (рисунок 2.9, б).

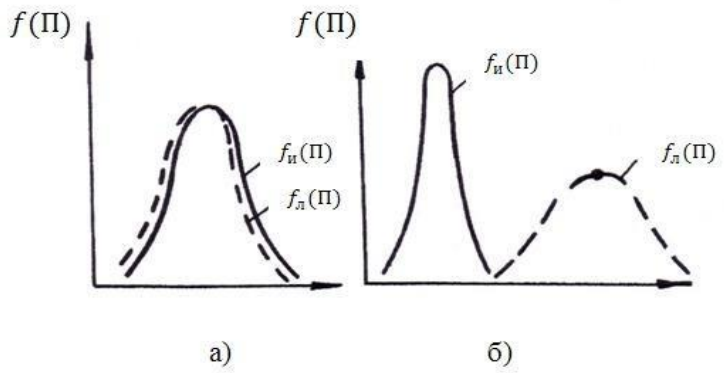


Рис. 2.9 – Щільності розподілу ознак розпізнавання: а) при абсолютній подобі; б) при абсолютній відмінності.

Таким чином, діапазон зміни показників для ймовірності розпізнавання (0,5…1), для ймовірності не розпізнавання (0…0,5).

Зручним і наочним показником технічної подоби може служити середня частка нерозпізнаних об'єктів певного класу. Цей умовний показник являє собою відношення відповідної ймовірності не розпізнавання до максимально можливого її значення (при абсолютній подобі, коли можна вважати, що всі об'єкти не розпізнані), тобто:

$$P_{\text{и/л}}\Big/_{0,5} = 2P_{\text{и/л}};\ \delta^{\text{л}}_{\text{нр}} = P_{\text{л/и}}\Big/_{0,5} = 2P_{\text{л/и}}$$

$$P_{\text{и/л}}\Big/_{0,5} = 2P_{\text{и/л}};\ \delta^{\text{л}}_{\text{нр}} = P_{\text{л/и}}\Big/_{0,5} = 2P_{\text{л/и}} \qquad (2.5)$$

$$P_{\text{и/л}} = 0 \text{ и } P_{\text{л/и}} = 0) \quad \delta^{\text{и}}_{\text{нр}} = \delta^{\text{л}}_{\text{нр}} = 0,$$

При абсолютній подобі відображає фізичну сутність процесу розпізнавання.

У сучасних засобах розпізнавання образів для одержання радіолокаційних, інфрачервоних і інших синтезованих зображень об'єктів і оцінки їх ознак

використовуються методи глибокої обробки прийнятих сигналів у широкому спектрі діапазону хвиль. Для підвищення ефективності розпізнавання можуть бути використані багатоканальні перспективні оптико-електронні пристрої обробки інформації. Штучні нейронні мережі дозволяють із успіхом вирішувати проблеми ідентифікації, класифікації, керування, прогнозування, моделювання, оптимізації. Підвищення якості одержуваної інформації при цьому приводить до збільшення енергетичних, масо габаритних і вартісних показників цих пристроїв.

Завдання синтезу оптимального образа полягає в узгодженні й ув'язуванню раціональних технічних рішень по і-м підсистемам зображення, отриманих у результаті структурно-параметричного синтезу кращих варіантів, в обліку інтегративного ефекту їх взаємодії й рівня комплексування. Облік цих факторів може бути реалізований на базі математичних моделей, алгоритмів і програм і комплексного критерію.

Розв'язок усіх завдань структурно-параметричного синтезу являє собою ітераційний процес послідовних наближень. Проектувальник у процесі розв'язку уточнює, накопичує й обробляє інформацію. При цьому широко використовуються взаємодоповнюючі методи аналізу й синтезу, індукції (від загального до частки) і дедукції (від часткового до загального), а також формалізовані процедури прийняття рішень.

Завдання вибору раціонального устрою, що розпізнає, і-го типу, полягає в знаходженні технічних рішень і їх проектних параметрів з діапазону їх можливої зміни, яка, по-перше, задовольняло б заданому на проектування, ряду прийнятих обмежень і, по-друге, щоб обраний критерій прийняв мінімальне (максимальне) значення. Для розв'язку цих завдань необхідно провести структурно-параметричний синтез матриці сумісних альтернативних технічних рішень і їх проектних параметрів [3].

Для розв'язку завдань дискретної (комбінаторної) оптимізації структурних рішень може бути використаний метод перебору кращих альтернатив і його модифікації – метод галузей і границь (спрямованого

перебору) і інші процедури. Більш докладно методи оптимізації викладені в роботах [3, 6].

Висновки по 2-му розділу

1. Виконана постановка завдання по створенню й удосконаленню системи розпізнавання обличчя у відеопотоках. Розроблена структура оптико-електронної системи виявлення й розпізнавання обличчя.

2. Сформульований алгоритм розпізнавання об'єктів зовнішнього середовища.

3. Розроблена узагальнена блок-схема алгоритму обробки кадрів розробленою системою.

4. Виконане дослідження можливостей практичного застосування методу Віоли-Джонса для розпізнавання обличчя.

5. Виконаний аналіз ефективності оптико-електронної системи розпізнавання об'єктів.

6. Запропонований байесовский метод оцінки ймовірності приналежності об'єкта до класу істинних і хибних.

3. ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ СИСТЕМИ РОЗПІЗНАВАННЯ ОБЛИЧЧЯ ЛЮДИНИ

3.1. Виявлення обличчя методом Віоли-Джонса

Сформулюємо основні положення, на яких заснований даний метод:

- інтегральна вистава зображень;
- пошук обличчя за допомогою ознак Хаара;
- каскадна класифікація із застосуванням бустингу.

Розглянемо більш докладно кожне із цих положень.

Інтегральна обробка зображень

Для розрахунків яскравості прямокутної ділянки зображення використовується так зване інтегральна обробка [18]. Дана обробка застосовується й у багатьох інших розроблених алгоритмах комп'ютерного зору. Інтегральна вистава дозволяє швидко розраховувати сумарну яскравість довільного прямокутника на заданому зображенні, причому час розрахунку не залежить від площі прямокутника.

Інтегральна вистава зображення являє собою матрицю, розміри якої збігаються з розмірами вихідного зображення. У кожному елементі такої матриці зберігається сума інтенсивності усіх пікселів, що перебувають лівіше й вище даного елементу.

Елементи матриці розраховуються у відповідності з наступною формулою:

$$I(x,y) = \sum_{x' \le x, y' \le y} i(x', y') \tag{3.1}$$

де $I(x, y)$ – значення крапки (x, y) інтегрального зображення; $i(x, y)$ – значення інтенсивності вихідного зображення.

Застосування інтегральної вистави зображення дозволяє обчислювати ознаки однакового виду, але геометричні параметри, що мають різні, за однаковий час, тому що розрахунок матриці інтегрального виразу забирає лінійний час, пропорційне числу пікселів у зображенні.

Ознаки Хаара

Ознакою f об'єкта a називають відображення $f: A \rightarrow df$, де Df - безліч припустимих значень ознаки. Якщо заданий набір векторів $f_1, \ldots, f_n$, тоді вектор $x=(f_1(a), \ldots, f_n(a))$ називається ознаковим описом об'єкту $a \in A$.[19]

Уперше використання для виявлення об'єктів ознак, заснованих на вейвлетах Хаара, було запропоновано в роботі Папагеоргиу в 1998 році [20]. Віола й Джонс адаптували цю ідею у своїй роботі й одержали прямокутні ознаки, названі ознаками Хаара [3]. Зовнішній вигляд даних ознак можна побачити на рисунку 3.1.

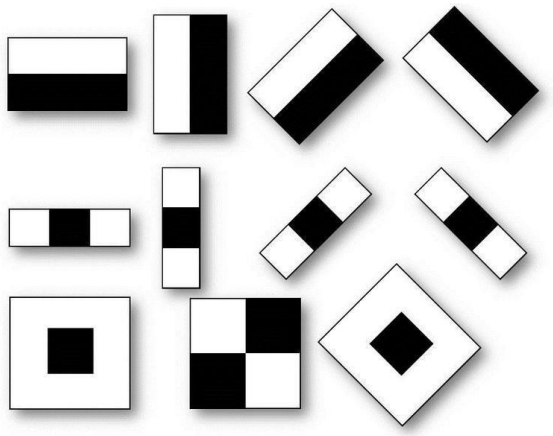


Рис. 3.1 – Ознаки Хаара

У розширеному методі Віоли-Джонса, представленому в бібліотеці комп'ютерного зору Opencv, і використовуваному в розроблювальній системі, використовуються також додаткові ознаки, представлені на рисунок 3.2.

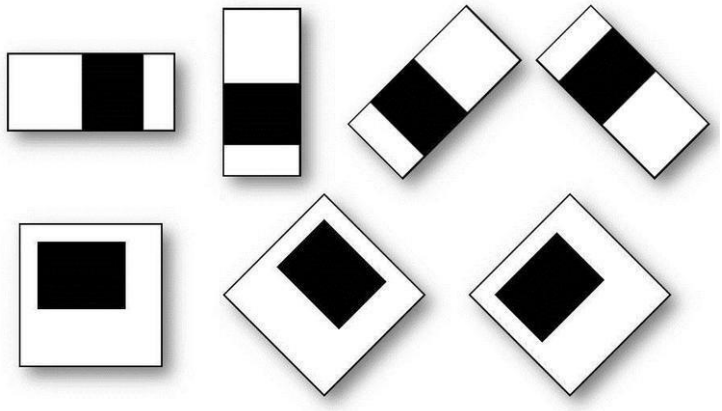


Рис. 3.2 – Додаткові ознаки Хаара

Результатом обчислення такої ознаки на інтегральній вирази зображення буде:

$$F = U - V \tag{3.2}$$

де U – сума значень крапок, що закриваються світлою частиною ознаки, а V – сума значень крапок, що закриваються темною частиною ознаки. Такі ознаки описують опис перепаду яскравості по обом осям зображення.

Пошук обличчя відбувається за допомогою так званого скануючого вікна, розміри якого в оригінальному алгоритмі становлять 24x24 пікселя. Вікно переміщається по зображенню із кроком в 1 піксель і для кожного його положення обчислюються ознаки Хаара з різним масштабом і положенням у вікні. При цьому саме сканування проводиться також і для різних масштабів скануючого вікна. Знайдені ознаки передаються класифікатору, який визначає по їхніх значеннях, чи є область зображення, відповідна до вікна, особою чи ні.

Каскадна класифікація

Каскадна структура класифікатора дозволяє прискорити виявлення обличчя, фокусуючи роботу на найцікавіших областях зображення. Каскад являє собою структурну організацію слабких класифікаторів, навчених із застосуванням процедури бустингу. У такий спосіб при малих обчислювальних витратах можна на ранніх етапах розпізнавання відкинути зображення, з

великою часткою ймовірності не утримуючі шуканий об'єкт (у цьому випадку обличчя). Приклад каскадної структури класифікаторів представлений на рисунку 3.3.

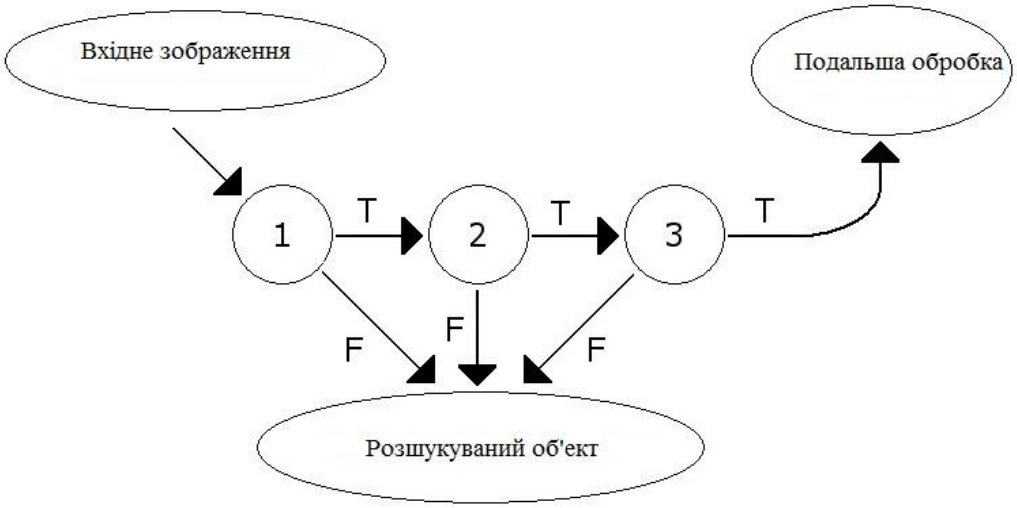


Рис. 3.3 – Каскадний класифікатор

Кожний рівень каскаду навчається за допомогою раніше згаданого алгоритму Adaboost [3]. Кількість особливостей, використовуваних у ньому, збільшується доти, поки виявлення цільового об'єкта й помилки першого роду не досягнуть заданого значення. Рівні визначаються шляхом тестування поточного детектора на безлічі, що перевіряє. Якщо загальна помилка першого роду для всього об'єкта ще не досягнута, то в каскад додається ще одна верства. Негативна безліч для навчання наступних верств виходить шляхом збору всіх неправильних виявлень при використанні поточного каскаду.

У результаті класифікації буде отриманий набір областей зображення, що містять шуканий об'єкт. Потім виключаються вкладені повторення при виявленні того самого об'єкта, викликані масштабуванням вікна.

Для подальшої обробки знайдені обличчя переводяться в градації сірого й масштабуються до розміру 128x128 пікселів.

3.2. Використання фільтра Гаусу для усунення шумів

З метою усунення шумів на зображеннях обличчя будемо використовувати фільтр Гауса. Фільтр Гаусу – це фільтр розмиття зображення, який використовує нормальний розподіл (також називане Гаусовим розподілом) для обчислення перетворення, застосовуваного до кожного пікселя зображення. Нормальний розподіл для двох вимірів описується наступним співвідношенням [21]:

$$G(u,v) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)} \tag{3.3}$$

де r – радіус розмиття; $r^2 = x^2 + y^2$; σ – стандартне відхилення розподілу Гаусу.

Дана формула задає поверхня, що має вид концентричних окружностей з нормальним розподілом від центральної крапки. Пікселі, де розподіл відмінний від нуля використовуються для побудови матриці згортки, яка застосовується до вихідного зображення. Значення кожного пікселя стає середньо зваженим для околиці. Вихідне значення пікселя ухвалює найбільшу вагу (має найвище Гаусово значення), і сусідні пікселі ухвалюють менші ваги, залежно від відстані до них.

Приклад роботи розмиття по Гаусу для одномірного масиві можливо побачити на рисунку 3.4.

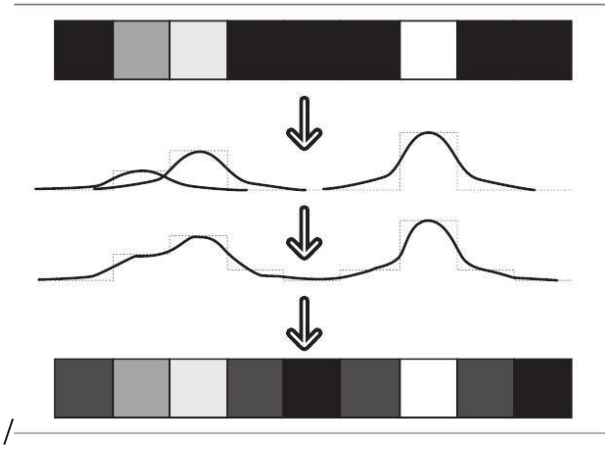


Рис. 3.4 – Розмиття по Гаусу на одномірному масиві

Таким чином, розмиття по Гаусу дозволить нам позбутися небажаних шумів на зображеннях, що зведе до мінімуму їх вплив при подальшій класифікації обличчя. Результат застосування фільтра до цілого зображення продемонстрований на рисунок 3.5.

Рис. 3.5 – Результат застосування фільтра Гаусу

3.3. LBP – перетворення

3.3.1. Аналіз можливостей використання LBP - перетворення в завданнях розпізнавання оптичних об'єктів

LBP- оператор уперше був запропонований в 1996 році для класифікації текстур [13]. Однак, пізніше знайшов застосування й для розпізнавання обличчя [14]. Суть цього оператора полягає в застосуванні до пікселів зображення граничного перетворення, у якому значення яскравості оброблюваного пікселя рівняється зі значеннями яскравості пікселів його околиці. Результат порівняння кожного пікселя околиці з оброблюваним пікселем переводиться у двоїчне число.

У класичному варіанті використовується квадратна околиця розміром 3x3 пікселя. Приклад розрахунків LBP- перетворення для такої околиці наведений на рисунку 3.6.

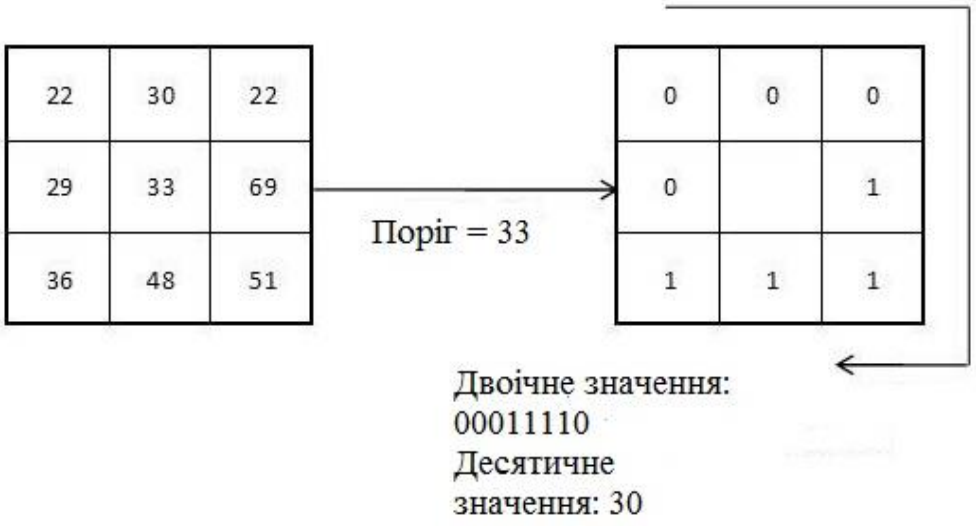


Рис. 3.6 – Класичний LBP- оператор

Після застосування LBP- оператора, зображення ділиться на прямокутні області, для кожної з яких розраховуються гістограми, що описують, наскільки часто зустрічаються в даній області пікселі різних значень яскравості. Значення елементів LBP- гістограми можуть бути описані наступною формулою:

$$H_i = \sum_{x,y} I\{f(x,y) = i\}, i = 0, \dots, n-1 \quad (3.4)$$

де *f(x, y)* – значення яскравості пікселя LBP зображення з координатами (*x, y*); *n* – кількість різних значень яскравості пікселів; $I\{A\} = 1$, якщо A – правда, інакше $I\{A\} = 0$.

Отримані гістограми нормалізуються, та використовуються надалі в якості ознак класифікації. В оригінальному дослідженні для класифікації використовується метод найближчого сусіда, який докладно буде описаний у наступних розділах. Приклад розбивки зображення на прямокутні області й формування гістограм можна побачити на рисунок 3.7.

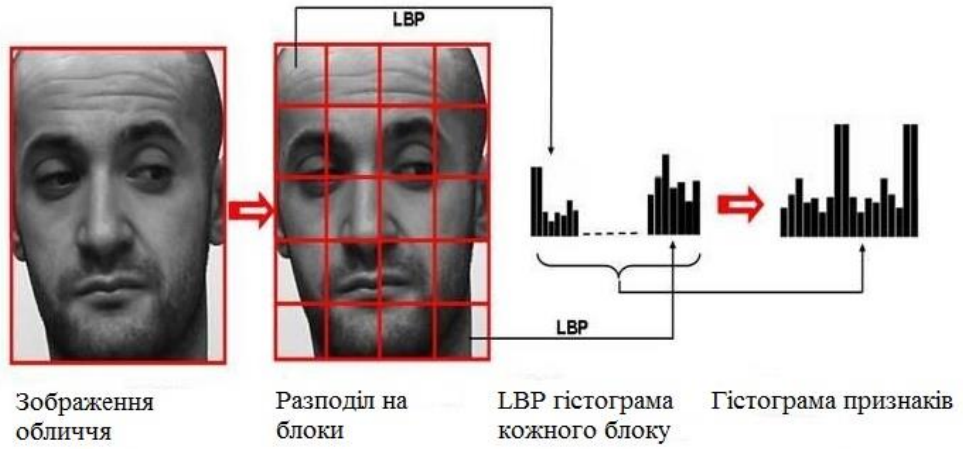


Рис. 3.7 – Розбивка зображення на прямокутні області й формування гістограми

У результаті виходить опис зображення обличчя в трирівневої локалізації. Слід зазначити, що такий опис не залежить від монотонних змін висвітлення [14].

Рівномірні локальні бінарні шаблони

Подальші дослідження локальних бінарних шаблонів показали, що істотну інформацію про форму об'єктів на зображенні несе тільки частина з них [21]. Такі локальні бінарні шаблони були названі рівномірними (uniform local binary patterns).

До даного виду LBP ставляться ті шаблони, двійковий код яких містить не більш двох переходів між нулем і одиницею. Вони описують тільки важливі локальні особливості зображення, такі як кінці ліній, грані, плями і т.д. Приклади рівномірних LBP представлені на рисунок 3.8.

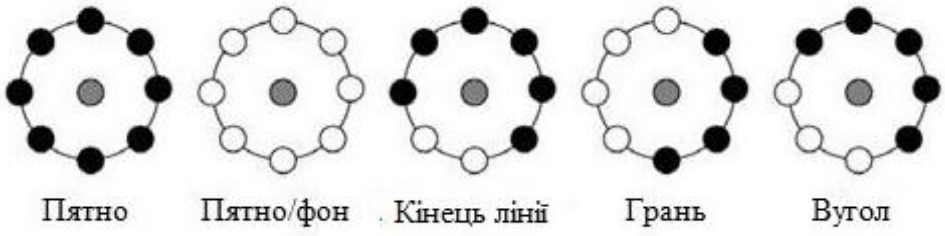


Рис. 3.8 – Рівномірні LBP

Усього налічується 58 рівномірних LBP. У результаті виходить 59-мірна гістограма ознак (додатковий розряд приділяється для підрахунку всіх нерівномірних LBP), на відміну від 256 - мірної гістограми в оригінальному алгоритмі. Таке скорочення розмірності дозволяє знизити витрати пам'яті й суттєво побільшати швидкість класифікації, при цьому поліпшивши її показники за рахунок використання тільки важливих ознак.

Симетричний - симетричні-центрально-симетричні LBP

Дана модифікація алгоритму розрахунків локальних бінарних шаблонів дозволяє ще сильніше скоротити витрати пам'яті й обчислювальну складність класифікації. Суть модифікації полягає в тому, що в якості граничного значення для кожного пікселя околиці ухвалюється не значення яскравості центрального пікселя околиці, а значення яскравості протилежного щодо центру околиці пікселя [23]. Порівняння розрахунків значення класичного LBP і центральносиметричного LBP наведене на рисунку 3.9.

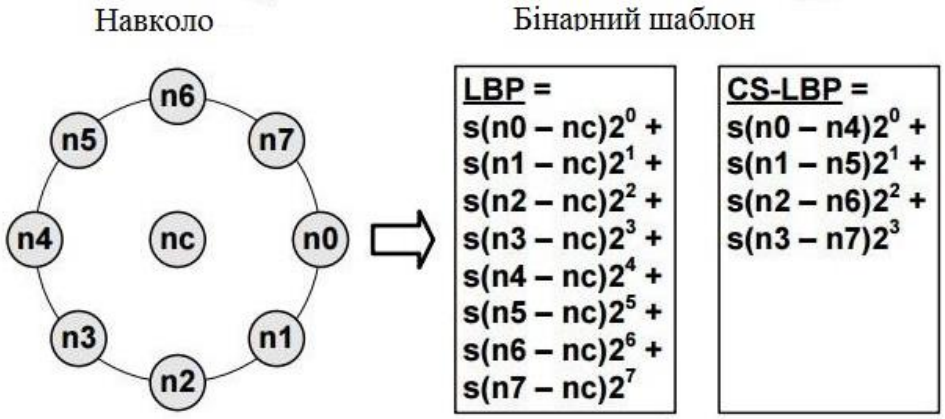


Рис. 3.9 – Розрахунки значення звичайного й центральносиметричного LBP (CS-LBP).

Як можливо бачити, у даному алгоритмі число розрядів значень перетворених пікселів скорочується до чотирьох. Відповідно розмірність центральносиметричного ознак скорочується до 24 = 16. У результаті виходить ще більша економія пам'яті й збільшення швидкості класифікації чому при

використанні рівномірних LBP. Дані переваги роблять дану модифікацію алгоритму практично ідеальним вибором для класифікації в реальному часі.

У цей час, таке скорочення розмірності гістограми ознак може негативно позначитися на точності класифікації. Відповідно при виборі ознаки класифікації необхідний пошук компромісу між швидкістю роботи й точністю системи.

3.3.2. Дослідження ефективності LBP – операторів

З погляду швидкодії ідеальним вибором у якості ознак класифікації при розробці системи розпізнавання в реальному часі є центральносиметричні локальні бінарні шаблони. Проте, перед застосуванням запропонованого алгоритму в розроблювальній системі має сенс переконатися в тому, що він не сильно програє більш повільним варіаціям LBP у точності класифікації.

Методика тестування

З метою розв'язання цього завдання всі три описані раніше алгоритми розрахунків LBP - гістограм були протестовані на двох різних наборах даних. При цьому також оцінювався найкращий варіант розбивки зображень на локальні області. У якості алгоритму класифікації використовувався метод найближчого сусіда. Створені в процесі дослідження реалізації LBP - перетворень і методу найближчого сусіда надалі минулому застосовані при розробці системи розпізнавання обличчя у відеопотоках, а також при тестуванні швидкості роботи цієї системи з використанням різних варіацій LBP-перетворення.

Першим набором даних, використаним для тестування була база зображень обличчя лабораторії Кембриджського університету [25]. Вона містила зображення 40 людей, по 10 зображень на кожний. Висвітлення на даних зображеннях не змінюється, однак присутні варіації в положенні обличчя при зйомці. Зображення одного з обличчя даної бази представлені на рисунку 3.10.

Рис. 3.10 – Зображення з бази даних Кембриджського університету

У якості навчальної й тестової вибірки використовувалося по 5 зображень на кожна обличчя. Тестова й навчальна вибірка не мали перетинань, однак для одержання результатів на більш великому масиві даних тестова й навчальна вибірки мінялися місцями, і тестування проводилося повторно. Перед обробкою зображення масштабувалися до розміру 128x128 пікселів. У результаті всього було класифіковано 400 зображень.

Другий набір даних – база зображень обличчя лабораторії Їєлського університету [26]. Дана база містить зображення 38 людей, по 65 зображень на кожний різні варіації, що включають, висвітлення. З них для тестування було відібрано по 10 зображень на кожну людину. Приклад зображень із другої бази обличь представлений на рисунку 3.11.

Рис. 3.11 – Зображення з бази даних Йельского університету

Варто відзначити, що в даній базі облич були кадровані точніше й у більшому масштабі, чому на зображеннях першого набору даних. Тестування проводилося аналогічно тестуванню першого набору.

Усього було досліджено 380 зображень.

Результати дослідження

Ефективність розпізнавання для кожного із трьох LBP операторів при тестуванні на першому наборі тестових даних представлена в таблиці 3.1.

Таблиця 3.1 – тестування LBP операторів на першому наборі даних

Розбивка / Метод	1x1	2x2	3x3	4x4	5x5	6x6	7x7	8x8
LBP	82,5%	91%	94%	95,5%	94%	93,25%	92,5%	89,3%
Uniform LBP	81%	93,8%	97%	94,5%	92%	92%	92%	89,5%
CS-LBP	67,8%	92,3%	95%	94,8%	94,3%	93,3%	90,3%	90,3%

Ефективність розпізнавання для кожного із трьох LBP операторів при тестуванні в другому наборі даних представлена в таблиці 3.2.

Таблиця 3.2 – Тестування LBP операторів на другому наборі даних

Розбивка / Метод	1x1	2x2	3x3	4x4	5x5	6x6	7x7	8x8
LBP	41,8%	71,6%	88,2%	91,8%	92,6%	93,2%	95,8%	96,1%
Uniform LBP	41,3%	75,26%	91,8%	91,8%	92,9%	92,1%	95%	94,2%
CS-LBP	20,3%	61,3%	84,5%	89,2%	89,2%	91,6%	92,6%	93,9%

Як можливо бачити з отриманих результатів, класичний LBP і Uniform LBP працюють приблизно з однієї точністю. Для досягнення точності 90 % і більш доцільно використовувати розбивки зображення починаючи від 4x4. Однак варто відзначити, що на першому наборі даних найкраще себе показала розбивка 3x3.

Центральносиметричний LBP при розбивці зображення на мале число блоків уступає іншим локальним бінарним шаблонам. Але при використанні

більшого числа під областей його показник точності класифікації відстає від інших LBP у середньому не більше ніж на 3 %. При тестуванні ж на першому наборі даних CS-LBP і зовсім перевершує інші шаблони на ряді розбивок.

У результаті можна сказати, що центральносиметричні локальні бінарні шаблони доцільно використовувати в системі розпізнавання обличчя у відеопотоках, через високу швидкість роботи й показників точності, що майже не уступають іншим LBP. Оптимальним по співвідношенню точності й витрат пам'яті розбивкою зображення на під областю при використанні CS-LBP є розбивка 4x4, яке й буде використовуватися в розроблювальній системі. Дана розбивка забезпечує стабільно високий відсоток вірних класифікацій при не дуже більших витратах пам'яті.

3.4. Метод застосування маски значимих областей зображення

Зображення обличчя, одержувані після процедури виявлення, мають квадратну форму. Однак обличчя займає не весь простір такого зображення. Тому логічно було б виключити вплив на розв'язок класифікатора областей зображення, у яких немає обличчя.

Простим способом розв'язку даної проблеми є застосування маски значимих областей зображення. Така маска являє собою зображення одного розміру з оброблюваним зображенням. Пікселі ненульової яскравості в масці відповідають значимим областям. У нашому випадку значимою областю є овальна область у центрі зображення, відповідного обличчя.

Значення пікселів результуючого зображення можна знайти, користуючись наступним математичним вираженням:

$$R(x,y) = \begin{cases} I(x,y) & \text{если } M(x,y) \neq 0 \\ 0 & \text{если } M(x,y) = 0 \end{cases} \quad (3.4)$$

де $I(x, y)$ – значення яскравості пікселя оброблюваного зображення; $M(x, y)$ – значення яскравості пікселя маски значимих областей.

При розв'язку завдання класифікації з використанням локальних бінарних шаблонів маску значимих областей доцільно застосувати після виконання LBP- перетворення й перед розрахунками гістограм. Таким чином, усі незначущі пікселі зображення на гістограмі будуть згруповані в одне значення. Приклад застосування маски значимих областей до зображення, обробленого LBP- оператором, наведені на рисунку 3.12.

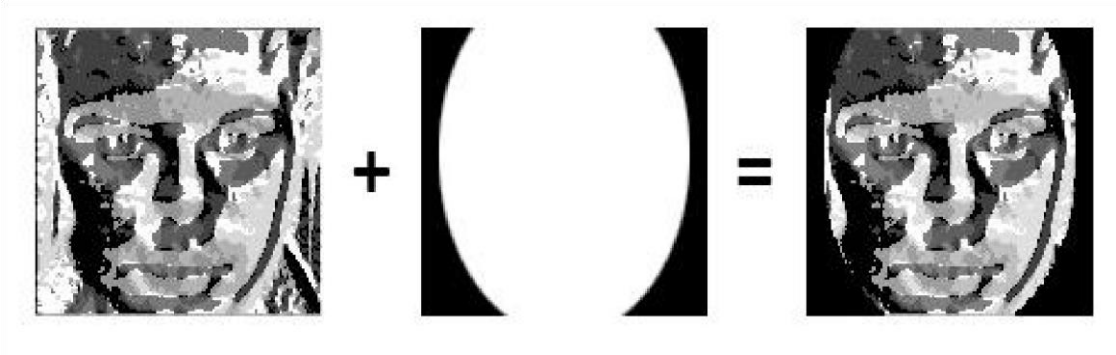


Рис. 3.12 – Застосування маски значимих областей

Як бачимо з рисунка 3.12, у результаті застосування даної операції пікселі зображення, які не повинні впливати на результат класифікації, ухвалюють нульове значення яскравості.

3.5. Класифікація LBP- гістограми методом найближчого сусіда

Отримані LBP - гістограми можуть бути класифіковані за допомогою методу найближчого сусіда, як і в оригінальному дослідженні LBP стосовно до завдання розпізнавання обличчя [14]. Даний метод є, простим алгоритмом класифікації, суть якого полягає в тому, що об'єкт ставиться до того класу, до елемента якого він ближче всього перебуває. Наприклад, на рисунку 3.13 зелене коло відповідно до даного алгоритму повинно бути класифіковане як червоний трикутник.

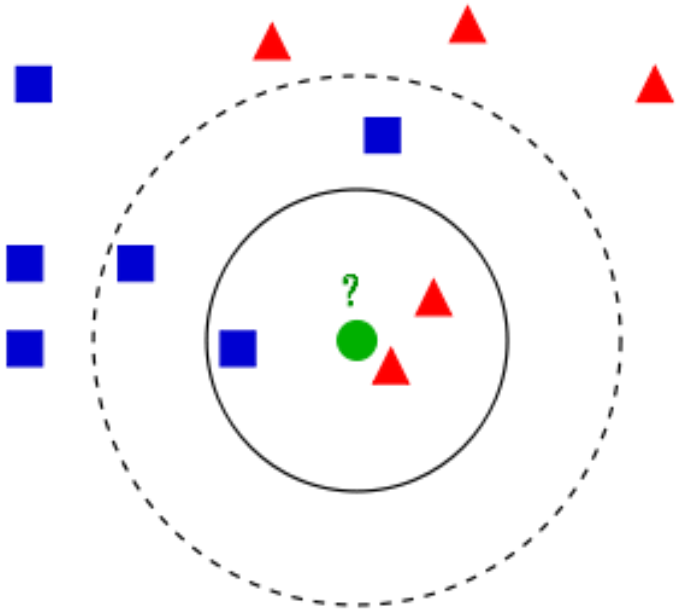


Рис. 3.13 – Метод найближчого сусіда

Для поліпшення результатів також використовують метод, у якому об'єкт відносять до того класу, до якого ставиться більшість його сусідів в околиці заданого розміру. Однак дослідження, проведене в рамках даної роботи, показало, що при розв'язку завдання класифікації обличчя такий підхід негативно впливає на роботу класифікатора.

Математично алгоритм можна описати в такий спосіб. На першому кроці визначається елемент x_s навчальної вибірки з N елементів, який ближче всього до пред'явленого образа x, тобто:

$$\|x - x_s\| = \min\{\|x - x_i\| : i = 1, \dots, N\} \quad (3.5)$$

На другому кроці перевіряється умова приналежність до класу: якщо $x_s \in$ ώj, те тоді вважається що й $x \in$ ώ$_j$.[26]

Метод найближчого сусіда застосовують у тому випадку, коли ціна помилки неправильної класифікації є великий, а помилки даних невеликі. Основним його недоліком є істотна чутливість до значень окремих (можливо помилкових) даних. Незважаючи на це даний метод показує високу ефективність при застосуванні в широкому спектрі завдань класифікації [24].

Особливої уваги також заслуговує питання вибору метрики, що визначає відстань між гістограмами. Для досягнення максимальної точності класифікації

необхідно вибрати ту метрику, яка найбільше адекватно б відображала відмінності між гістограмами зображень різних класів. В оригінальному дослідженні [14] застосовується так зване відстань Chi-Square, що розраховується по наступній формулі:

$$\sum_{i=1}^{n} \frac{(x_i \quad y_i)^2}{(x_i + y_i)} \tag{3.6}$$

де x_i – і-е значення першої гістограми; y_i – і-е значення другої гістограми.

3.6. Розробка системи розпізнавання обличчя

Розробка системи виконана об'єктно-орієнтованою мовою програмування C# у середовищі розробки Microsoft Visual Studio 2013. C# ставиться до родини мов з C-Подібним синтаксисом, з них його синтаксис найбільш близький до C++ і Java. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (у тому числі операторів явного й неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи й методи, ітератори, анонімні функції з підтримкою замикань, LINQ, виключення, коментарі у форматі XML [27].

Наявність механізму «збору сміття» в C++ дозволить ефективно й просто організувати роботу з обліковими структурами, на яких заснована робота системи. Немаловажним критерієм на користь вибору даного мови програмування для розробки є досвід розробки на ньому, отриманий за час навчання.

З метою спрощення процесу розробки було вирішено використовувати бібліотеку Opencv. Дана бібліотека розроблена на C/C++, а також має інтерфейси для Python, Java і інших мов, у тому числі й для .NET мов – Emgucv, яка й була використана в роботі. Підтримує Windows, Linux, Mac OS, ios і Android. Вона містить алгоритми для обробки, реконструкції й очищення зображень, розпізнання образів, захоплення відео, спостереження за об'єктами, калібрування

камер і ін. Бібліотека поширюється по ліцензії BSD, а виходить, може вільно використовуватися в академічних і комерційних цілях [28, 29].

Наявність великої кількості реалізованих алгоритмів комп'ютерного зору, а також великий набір і статей теоретичних матеріалів по їхнім застосуванню роблять бібліотеку Opencv ідеальним рішенням для використання в проектах, присвячених вирішенню проблем комп'ютерного зору.

Аналіз потоків даних

Відповідно до загального алгоритму обробки кадрів відеопотоку можна скласти діаграму потоків даних розроблювальної системи,

Дана діаграма представлена на рисунку 3.13. Виходячи з діаграми потоків даних, доцільно розробити класи, відповідні до сутностей, що берете участь у перетворенні даних. До таких сутностей можна віднести детектор обличчя і LBP- перетворювач. Класифікатор має сенс реалізувати методом головного класу додатка, оскільки він буде тісно взаємодіяти зі списками розпізнаних і нерозпізнаних обличчя, що є атрибутами головного класу. Крім цього, необхідно реалізувати функцію розрахунків гістограм, клас, що описує категорію обличчя, і клас, що описує розпізнане обличчя.

На вхід детектора надходять кадри відеопотоку, на яких він шукає обличчя. У результаті виходить список прямокутних областей кадру, відповідних до положення на ньому обличчя. Сам каскадний класифікатор реалізується коштами бібліотеки Emgu CV, у якій є його реалізація з набором якісно навчених каскадів. Облікові структури будуть описуватися коштами бібліотеки колекцій мови C++.

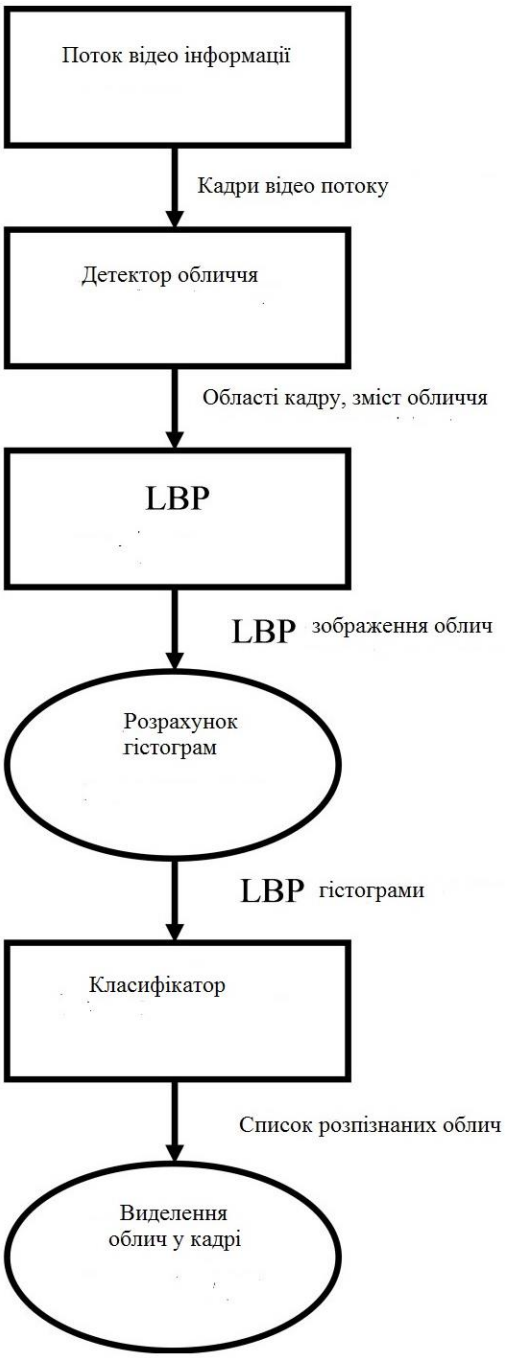


Рис. 3.13 – Діаграма потоків даних системи

LBP- перетворювач повинен реалізовувати алгоритм перетворення зображень центральносиметричним LBP- оператором. На вхід перетворювача будуть подаватися зображення обличчя. Вихідними даними є перетворені LBP-зображення.

Клас головної форми додатка буде містити елементи інтерфейсу додатка, а також список розпізнаних і список нерозпізнаних обличчя, якими буде оперувати система при обробці кадрів. Крім цього даний клас реалізує функції розрахунків гістограм і графічного відображення результатів роботи системи.

Діаграма класів додатку наведена у Додатку на рисунку 3.14.

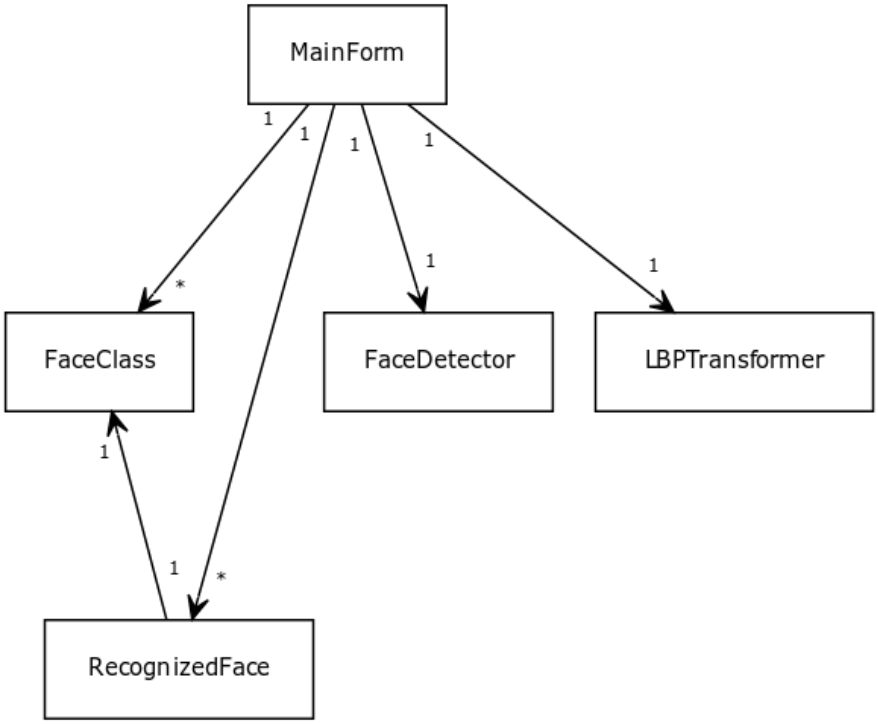


Рис.. 3.14 – UML- діаграма класів додатку

З міркувань компактності, у представленій діаграмі відбиті тільки асоціативні зв'язки класів без їхнього вмісту. Головний клас додатка Mainform містить список класів обличчя типу Faceclass і список розпізнаних обличчя типу Recognizedface. Так само він використовує у своїй роботі детектор обличчя Facedetector і LBP перетворювач зображень Lbptransformer. Кожна розпізнана обличчя Recognizedface при цьому посилається на відповідний йому клас обличчя Faceclass.

Вихідний код представлених класів мовою програмування C# з докладними коментарями представлений у додатку.

Recognizedface

Клас Recognizedface описує розпізнане класифікатором обличчя. Елементи даного класу зберігаються в обновлюваному списку розпізнаних обличчя, на основі даних якого проводиться відображення інформації на формі.

Атрибути класу:

private Faceclass _faceclass – клас обличчя, до якого належить дане розпізнане обличчя.

private Rectangle _rect – прямокутна область, що відповідає положенню обличчя в кадрі відеопотоку.

private Mat _histogram – LBP гистограмма обличчя.

private Image<Gray, Byte> _face – зображення обличчя, узяте з оброблюваного в теперішній момент кадра відеопотоку.

private Image<Gray, Byte> _lbp – LBP вистава зображення face.

private double _distance – дистанція між LBP гистограмой обличчя й LBP гистограммой найближчого елемента його класу.

Методи класу:

public Recognizedface(Faceclass fc, Rectangle r, Image<Gray, Byte> f, Image<Gray, Byte> l, Mat h, double d) – конструктор класу. У якості параметрів передаються значення, що привласнюються атрибутам створюваного об'єкту.

public void Setfaceclass(Faceclass fc) – привласнює значення параметра атрибуту f*aceclass*.

public void Setrect(Rectangle r) – привласнює значення параметра атрибуту *rect*.

public void Sethist(Mat h) – привласнює значення параметра атрибуту *histogram*.

public void Setlbp(Image<Gray, Byte> l) – привласнює значення параметра атрибуту _lbp. *public void Setface(Image<Gray, Byte> f)* – привласнює значення параметра атрибуту *face*.

public void Setdist(double d) – привласнює значення параметра атрибуту *distance*.

public Faceclass Getfaceclass() – повертає значення атрибута *faceclass*.

public Rectangle Getrect() – повертає значення атрибута *rect*. *public Mat Gethist()* – повертає значення атрибуту *histogram*.

public Image<Gray, Byte> Getlbp() – повертає значення атрибуту *lbp*. *public Image<Gray, Byte> Getface()* – повертає значення атрибуту *face*.

public double Getdist() – повертає значення атрибуту *distance*.

LBP – transformer

Даний клас є статичним класом, не має атрибутів і містить тільки один метод, що виконує центральносиметричне LBP- перетворення зображення. Клас LBP - transformer описує CS-LBP перетворювач. Має на увазі доповнення іншими методами LBP- перетворення при подальшому розвитку розробки.

Методи класу: *public static Image<Gray, Byte> Cstransform(Image<Gray, Byte> input)*

- здійснює LBP- перетворення вхідного зображення input і повертає перетворене зображення.

Facedetector

Клас Facedetector описує детектор обличчя, що використовує у своїй роботі метод Віоли-Джонса.

Атрибути класу: *private Cascadeclassifier haar* – каскадний класифікатор, за допомогою якого проводиться виявлення обличчя.

Методи класу:

public Facedetector() – конструктор класу. Завантажує в атрибут *haar* навчений каскад для розпізнавання обличчя у вигляді *xml* файлу.

public Rectangle[] Getfacesrect(Image<Bgr, Byte> frame, double scalefactor, int minneighbors, int sz) – одержує на вхід зображення frame, на якому проводиться пошук обличчя, а так само дані для настроювання параметрів виявлення, а саме фактор збільшення скануючого вікна scalefactor, мінімальна кількість вкладених виявлень minneighbors і мінімальний розмір обличчя sz. Повертає список прямокутників, відповідних до положень обличчя на зображенні frame.

Варто відзначити, що мінімальна кількість вкладених виявлень прямо впливає на точність виявлення обличчя.

Даний параметр задає необхідне для визнання області зображення особою кількість спрацьовувань детектора при різних масштабах скануючего вікна, що змінює свої розміри відповідно до фактору збільшення scalefactor, у даній області зображення.

Сам процес виявлення обличчя здійснюється за допомогою функції Cascade Classifier Detect Multiscale бібліотеки Emgu CV, викликуваної для каскадного класифікатора *haar*. Дана функція являє собою реалізацію каскадної класифікації з методу Віоли-Джонса. Їй же й передаються, описані вище параметри.

Faceclass

Даний клас описує клас (категорію) обличчя. Кожний клас обличчя відповідає конкретній розпізнаваній людині.

Атрибути класу:

private Image<Gray, Byte> img – зображення обличчя, що використовується разом з назвою класу для зручності ідентифікації й розрізнення класів.

private Mat[] _histogram – масив LBP- гістограм різних зображень обличчя, відповідного до класу. Використання не однієї, а декількох гистограмах, продиктоване особливостями методу найближчого сусіда, оскільки при

розпізнаванні програма шукає найбільш схожого представника класу із усіх наявних.

private String _name – ім'я класу обличчя.

Методи класу: *public Faceclass(Image<Gray, Byte> faceimg, Mat[] facehist, String facename)* – конструктор класу. У якості параметрів передаються значення, що привласнюються атрибутам створюваного об'єкта.

public Mat[] Gethist() – повертає значення атрибуту *histogram*. *public String Getname()* – повертає значення атрибута *name*.

public Image<Gray, Byte> Getimg() – повертає значення атрибуту *img*.

Mainwindow

Клас Mainwindow описує головне вікно додатка, а так само реалізує головний модуль додатку.

Атрибути класу:

private Facedetector _detector – детектор обличчя.

private List<Faceclass> _faces – список класів обличчя, що зберігаються в пам'яті програми в поточний момент. *private List<Recognizedface> recognizedfaces* – список обличчя, розпізнаних у поточному кадрові відеопотоку.

private List<Rectangle> notrecognized – список областей поточного кадра відеопотоку обличчя, що містять, які класифікатор не може віднести до якого-обличчя з наявних класів.

private Image<Gray, Byte> _mask – зображення маски значимих областей.

private Image<Gray, Byte>[] _currentfaces – зображення обличчя, зняті користувачем для формування нового класу обличчя.

private int _currentfacescount – поточне число зображень обличчя для формування нового класу. *private Rectangle[] _facesrect* – список областей поточного кадра відеопотоку обличчя, що містять.

private bool _capturing – прапор, що показує проводиться чи в даний момент захоплення відеопотоку.

private Capture _capture – об'єкт, що здійснює захоплення відеопотоку.

private String videopath – шлях до відеофайлу *private Mat frame* – поточний кадр відеопотоку.

private Image<Bgr, Byte> frameimg – також поточний кадр відеопотоку.

Необхідний для здійснення ряду операцій, неможливих з використанням об'єкта типу Mat.

Методи класу:

public Mainform() – конструктор класу.

public Mat Calchistogram(Image<Gray, Byte> image) – повертає гистограмму зображення *image.*

private void Recognizeface(Rectangle rect, Image<Gray, Byte> face, Image<Gray, Byte> lbp, Mat histogram, decimal threshold) – метод здійснюючий розпізнавання обличчя і формування списків розпізнаних і нерозпізнаних обличчя.

У якості параметрів методу передаèтся зображення обличчя face, що відповідає йому область кадра rect, LBP - зображення обличчя LBP і його гістограма *histogram*, а також поріг розпізнавання threshold. На основі цих даних і списку класів обличчя виконується класифікація обличчя методом найближчого сусіда. Блок-схема реалізації методу найближчого сусіда у функції Recognizeface представлена на рисунку 3.15.

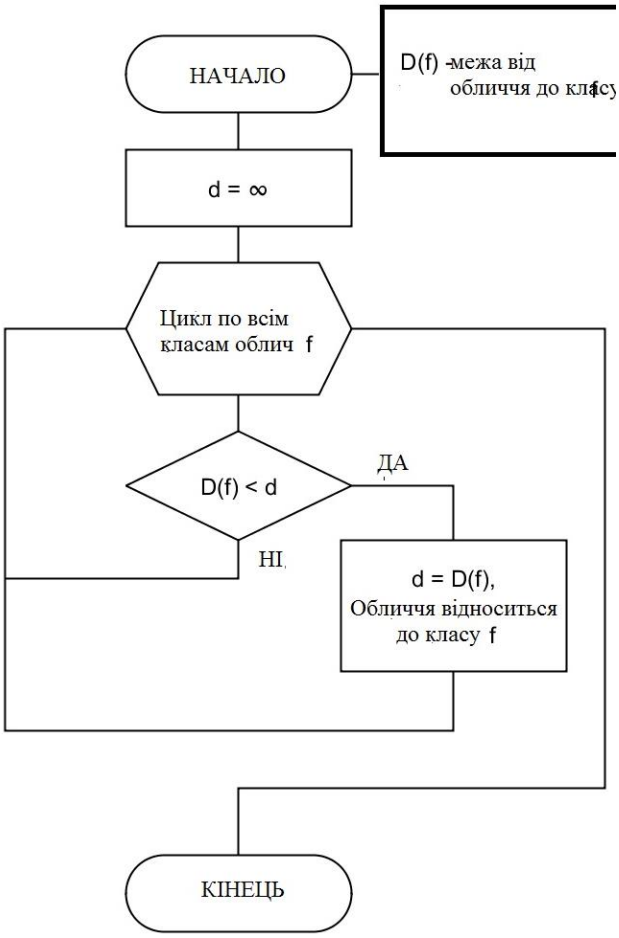


Рис. 3.15 – Блок-схема реалізації розпізнавання обличчя методом найближчого сусіда

Після визначення класу обличчя відбувається порівняння відстані до класу із граничним значенням, у результаті чого обличчя або міститься в список розпізнаних облич, або в список нерозпізнаних обличчя. Блок-схема алгоритму формування списку розпізнаних облич представлена на рисунку 3.16.

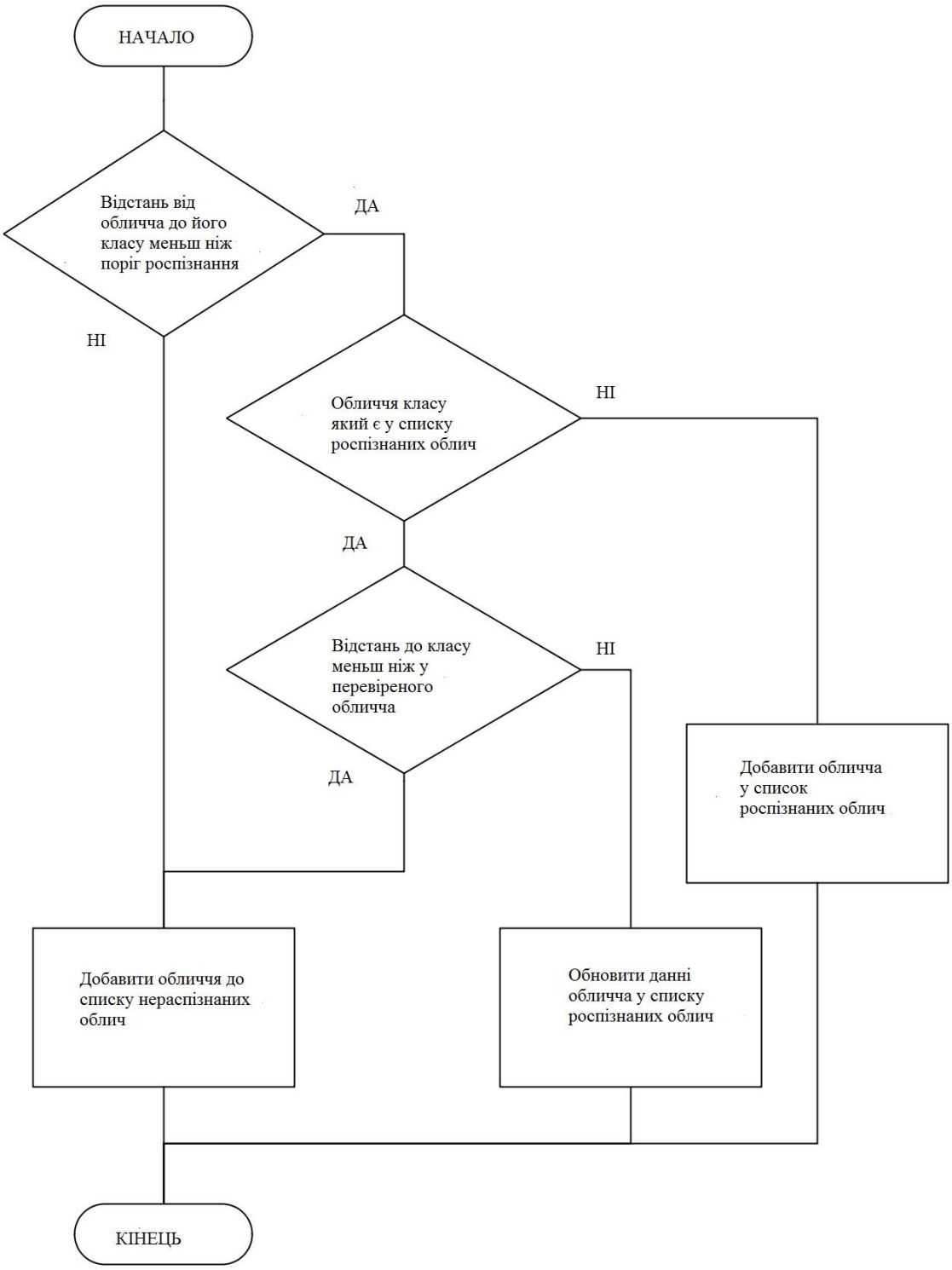


Рис. 3.16 – Блок-схема алгоритму формування списків розпізнаних і нерозпізнаних обличь

private void Processfaces() – цей метод виконує обробку виявлених у кадрові відеопотоку обличчя відповідно до алгоритму, блок-схема якого представлена на рисунку. 3.17.

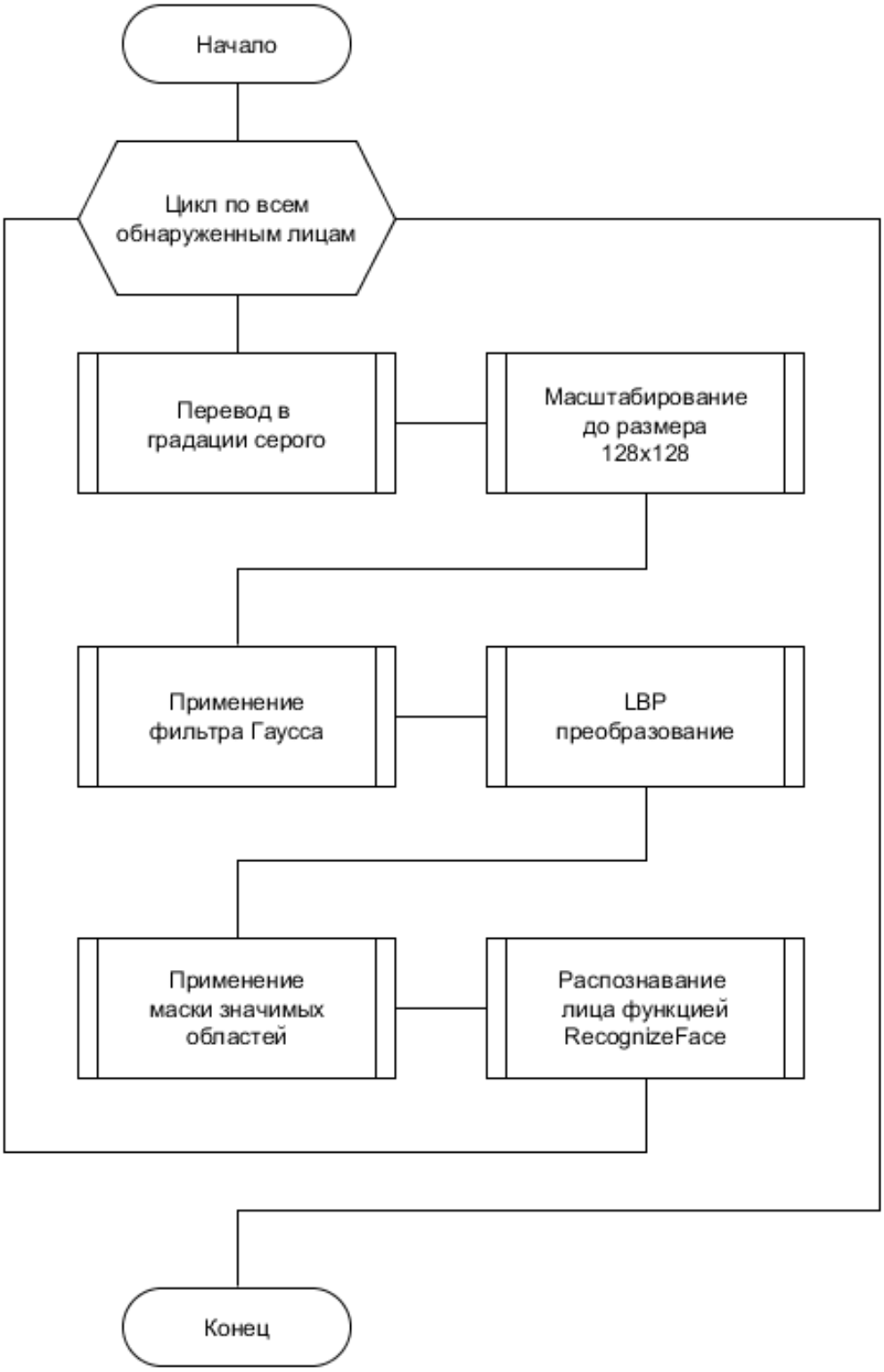


Рис. 3.17 – Блок-схема алгоритму обробки виявлених облич

private void Processframe(object sender, Eventargs arg) – функція обробки кадру відеопотоку. Працює відповідно до алгоритму, блок-схема якого представлена на рисунку 3.18.

Рис. 3.18 – Блок-схема алгоритму обробки кадрів відеопотоку

private void Drawdetected() - дана функція виконує візуальне виділення обличчя у кадрах відеопотоку й висновок інформації про обраного користувача обличчя на форму відповідно до алгоритму, представленого на рисунку 3.19.

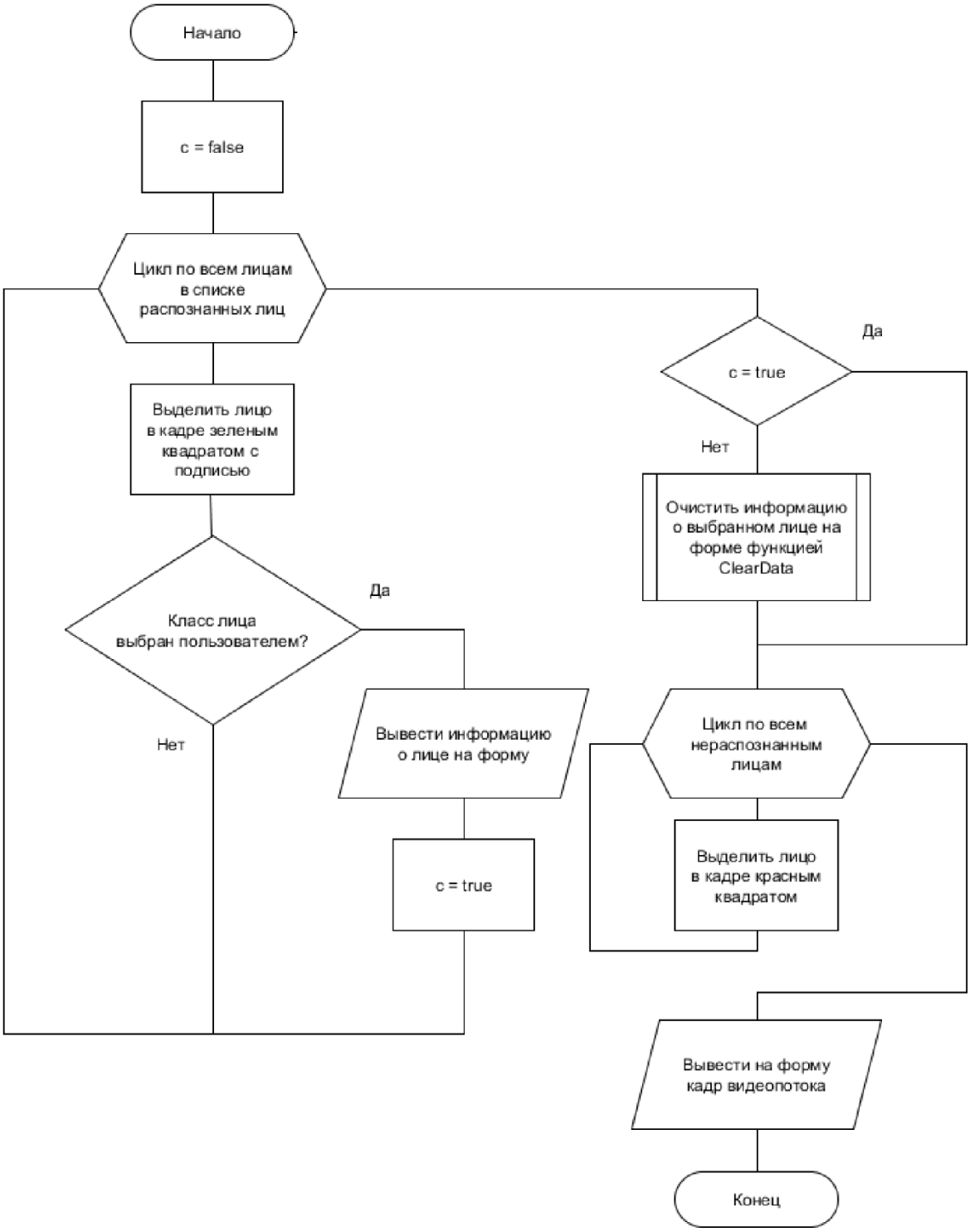


Рис. 3.19 – Блок-схема алгоритму одержання інформації про форму обличчя

private void Cleardata() – метод видаляє з форми інформацію про обраний користувачем обличчя. Використовується щораз при відновленні кадру відеопотоку перед завантаженням нової інформації про обличчя.

private void Drawhistogram(Mat histogram) – метод малює на спеціальному елементі форми гістограм histogram. *private void*

openfaceclass(object sender, Canceleventargs e) – метод завантаження в додаток класу обличчя із зовнішнього файлу. *private void addfacebutton_Click(object sender, Eventargs e)* – метод додає наявне в даний момент у кадрові обличчя в список зображень, використовуваний для формування нового класу обличчя.

Спрацьовує по натисканню кнопки Add Face на формі.

private void addfacebutton_Click(object sender, Eventargs e) – метод додає наявне в даний момент у кадрові обличчя в список зображень, використовуваний для формування нового класу обличчя.

Спрацьовує по натисканню кнопки Add Face на формі.

private void addclassbutton_Click(object sender, Eventargs e) – метод додає до списку класів обличчя новий клас, створений на основі зображень currentfaces. Спрацьовує по натисканню кнопки Add Class на формі.

private void faceclasseslistbox_Selectedindexchanged(object sender, Eventargs e) – метод, що спрацьовує при зміні обраного класу обличчя і вивідний на форму зображення нового обраного класу.

private void removefacebutton_Click(object sender, Eventargs e) – метод, що видаляє останнє зображення зі списку обличчя, використовуваних для створення нового класу. Спрацьовує по натисканню кнопки Remove Face на формі.

private void videobutton_Click(object sender, Eventargs e) – метод, що запускає зчитування кадрів з відеопотоку.

private void savefaceclass(object sender, Canceleventargs e) – метод, що виконує збереження обраного класу обличчя у файл. Використовує механізм повторювання.

private void saveclassbutton_Click(object sender, Eventargs e) – метод, викликуваний при натисканні кнопки Save Class. Відкриває діалогове вікно для збереження файлу.

private void openclassbutton_Click(object sender, Eventargs e) – метод, викликуваний при натисканні кнопки Open Class. Відкриває діалогове вікно для відкриття файлу.

private void removeclassbutton_Click(object sender, Eventargs e) – метод, що видаляє обраний клас обличчя зі списку. Спрацьовує по натисканню кнопки Remove на формі.

private void openvideobutton_Click(object sender, Eventargs e) – метод, викликуваний при натисканні кнопки Open File. Відкриває діалогове вікно вибору відео файлу.

private void openfiledialog2_Fileok(object sender, Canceleventargs e) – зберігає шлях до обраного відеофайлу в атрибут _videopath.

Опис інтерфейсу системи

При розробці інтерфейсу додатку можна обмежитися єдиним головним вікном, яке буде містити в собі активні елементи для настроювання параметрів роботи додатка, і області висновку даних про розпізнаваних обличчя.

Інтерфейс додатка містить кілька областей: область висновку обробленого відеопотоку з камери, область настроювання параметрів роботи додатка, область роботи із класами обличчя і область висновку інформації про обраний користувачем обличчя.

Розглянемо докладніше різні області інтерфейсу розробленої системи. Область висновку відеопотоку призначена для відображення оброблених кадрів. Крім цього дана область включає кнопки настроювання джерела відеопотоку, кнопку виклику діалогового вікна відкриття відеофайлу, і кнопку, що управляє початком і зупинкою захоплення відеопотоку.

У якості джерела відеопотоку може використовуватися підключена до комп'ютера веб-камеру або відеофайл. Для того, щоб вибрати відеофайл-джерело необхідно натиснути кнопку Open File. Після вибору файлу в діалоговому вікні, що з'явилося, і натискання кнопки ОК, шлях до обраного файлу відобразиться на формі. Запуск/зупинка обробки й висновку кадрів з обраного джерела здійснюється натисканням кнопки Start/Stop.

Область висновку інформації про обраний користувачем обличчя призначена для відображення даних розпізнаваних додатком обличчя. Джерелом

даних для відображення є обраний користувачем у списку Face Classes клас обличчя і розпізнана обличчя, відповідне до цього класу.

У даній області форми виводяться зображення обраного класу, прямокутна область поточного кадру, відповідна обличчя цього класу, а також LBP перетворене зображення обличчя. Крім цього відображається поточна відстань між гістограмою розпізнаного обличчя й гістограмою його класу й графічна вистава гістограми LBP зображення обличчя.

Область настроювань роботи додатка містить у собі дві групи настроювань. Перша група - це настроювання роботи розпізнавання обличчя. До них ставиться рівень розмиття по Гаусу й поріг розпізнавання. Поріг розпізнавання – максимально припустиме для ухвалення рішення про приналежність обличчя класу відстань між LBP- гістограмою обличчя й гістограмою класу.

Друга група настроювань даної області містить настроювання детектора обличчя. Значення даних настроювань передаються методу Getfacesrect детектора обличчя і були докладно описані в попередньому підрозділі.

Область настроювань роботи додатка представлений на рисунку 3.20.

Рис. 3.20 – Область настроювань роботи додатку

Область роботи із класами обличчя призначена для роботи зі списком класів обличчя, а також формування нових класів. У верхній частині області перебуває список завантажених у даний момент у програму класів обличчя. Користувач може вибирати потрібний йому клас натисканням лівої клавіші миші, і інформація про обраний клас і обличчях, йому відповідних, буде відображатися в області висновку інформації.

Нижче розташовані елементи інтерфейсу для формування нового класу обличчя. По натисканню кнопки Add Face у набір зображень для формування класу додається зображення обличчя, що присутнє зараз у кадрові відеопотоці. За допомогою кнопки Remove Face можна вилучити останнє зображення з набору. Як тільки необхідний набір зображень (до 30 включно) буде сформований, необхідно ввести ім'я класу в текстове поле й натиснути кнопку Add Class для створення на основі набору зображень нового класу обличчя. Створений клас відразу ж відобразиться в списку класів і буде використовуватися при розпізнаванні обличчя. Крім цього в даній області так само присутні кнопки для відкриття наявного класу обличчя з файлу, збереження обраного класу у файл і видалення обраного класу зі списку.

Область роботи із класами обличчя представлена на рисунку 3.21.

Рис. 3.21 – Область роботи із класами обличь

Також варто відзначити, що для запобігання помилок, різні елементи інтерфейсу системи включаються й вимикаються залежно від активності відеопотоку й наявності в ньому зображень обличчя.

3.7. Тестування роботи класифікатора обличчя

Тестування роботи класифікатора обличчя проводилося відповідно до умов, описаних у розділі, присвяченому аналізу завдання розпізнання. Захоплення відеопотоку проводилося за тих самих умов висвітлення, що й зйомка зображень навчальної вибірки.

Усього при тестуванні в програму було завантажено 20 класів обличчя. Для формування кожного класу використовувалося по 10 зображень. Із цих 20 класів облич 15 було створено з використанням зображень людей з мережі Інтернет. Дані класи обличчя були необхідні для того, щоб оцінити якість роботи класифікатора на великому масиві даних, а також, щоб провести тестування роботи програми при читанні відеопотоку з файлу.

Результати тестування показали, що точність роботи класифікатора при обробці відеопотоку становить більш 90 % вірно розпізнаних кадрів.

Однак варто відзначити чутливість розробленої системи до сильних немонотонних змін висвітлення, а також до змін положення й нахилу розпізнаваних обличчя, неврахованим при зйомці зображень для формування класів обличчя. При недотриманні необхідних умов правильної роботи додатка, точність класифікації помітно знижується.

Порівняння швидкості роботи LBP операторів

Вибір гістограм центральносиметричних локальних бінарних шаблонів як ознак класифікації був обумовлений високою швидкістю роботи й економією пам'яті при незначних втратах точності розпізнавання в порівнянні з іншими видами LBP операторів.

Дослідження проводилося для різних LBP операторів при різній кількості гістограм обличчя у базі. Швидкість обробки відеопотоку оцінювалася по кількості оброблюваних кадрів у секунду при розпізнаванні одного обличчя в кадрові. Результати дослідження зведені в таблиці 3.3.

Таблиця 3.3 – тестування швидкості роботи різних LBP- операторів

Гістограм у базі / МЕТОД	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
LBP	23	18	15	13	12	10	9	8	8	7
Uniform LBP	30	30	30	29	28	25	23	21	20	19
CS-LBP	30	30	30	30	30	30	30	30	30	30

Тестування проводилося на комп'ютері із процесором Intel core i5 с тактовою частотою 3,2 ГГц і об'ємом оперативної пам'яті 8 Gb. Дозвіл кадрів відеопотоку становив 640x480 пікселів. За результатами тестування, центральносиметричні локальні бінарні шаблони забезпечують обробку відеопотоку зі споконвічною частотою кадрів 30 кадрів у секунду навіть при наявності 10000 гістограм у базі облич. Інші дві варіації LBP- оператора показують значне погіршення продуктивності при подібному збільшенні числа гістограм у базі.

У результаті можна сказати, що центральносиметричні локальні бінарні шаблони є кращим вибором серед LBP- операторів для розв'язку завдання розпізнавання обличчя у відеопотоках у реальному часі.

Висновки по 3-му розділу

1. Описаний механізм роботи LBP- оператора і його різних модифікацій, а також принцип розрахунків LBP- гістограм і описана класифікація гістограм методом найближчого сусіда й принцип розрахунків відстаней між гістограмами.

2. Виконане дослідження ефективності розпізнавання обличчя при використанні трьох варіантів LBP- перетворення: класичного, рівномірного й центральносиметричного. Результати дослідження показали, що

центральносиметричний LBP- оператор практично не уступає в ефективності розпізнавання обличчя класичному й рівномірному LBP- операторам.

3. Зроблений висновок, що оптимальним вибором розбивки зображення на блоки по співвідношенню витрат пам'яті й ефективності розпізнавання є розбивка 4x4. Спроектовані й описані основні модулі розроблювальної системи розпізнавання обличчя.

4. Обраний інструментарій, необхідний для її розробки, а саме об'єктно-орієнтована мова програмування C#, бібліотека комп'ютерного зору Emgu CV і середовище розробки Microsoft Visual Studio2013.

5. Проведене тестування роботи розробленої системи й реалізованого в ній класифікатора обличчя. Отримані результат понад 90% вірних розпізнавань обличчя, при цьому зафіксоване погіршення результатів при недотриманні основних умов реєстрації зображення.

6. Проведене порівняння швидкостей обробки відеопотоку розробленою системою при використанні різних LBP- операторів, яке показало, що центральносиметричні локальні бінарні шаблони є кращим вибором серед LBP операторів для розв'язку завдання розпізнавання обличчя у відеопотоках у реальному часі.

ВИСНОВКИ

У роботі розроблено систему розпізнавання обличчя у відеопотоках на основі використання методу Віоли-Джонса і локальних бінарних шаблонів. Для виявлення обличчя у кадрах відеопотоку був використаний метод локальних бінарних шаблонів. Класифікація виявлених обличь проводилася методом найближчого сусіда з використанням центрально- симетричних локальних бінарних шаблонів. Тестування розробленої системи показало позитивні результати в приблизно 90 % вірних розпізнавань обличчя при обробці кадрів відеопотоку з вебкамери в реальному часі. Доведене, що гістограми центрально-симетричних локальних бінарних шаблонів є ефективним ознакою для класифікації обличчя у реальному часі. Розроблена система може використовуватися при рішеннях різних завдань відеоаналітики, і, у першу чергу, має безпосереднє застосування в системах контролю доступу та ідентифікації обличчя. При цьому невисокі системні вимоги роблять можливим застосування розробки на системах з низькою продуктивністю

1. Проведено дослідження ефективності різних модифікацій локальних бінарних шаблонів стосовно до завдання розпізнавання обличчя у реальному часі. Було встановлено, що центральносиметричні локальні бінарні шаблони практично не уступають іншим варіаціям LBP у якості ознак розпізнавання й при цьому мають набагато більш високу швидкість роботи. На підставі дослідження зроблений висновок, що гістограми центральносиметричних локальних бінарних шаблонів є ефективною ознакою для класифікації обличчя у реальному часі.

Тестування розробленої системи показало результати в приблизно 90 % вірних розпізнавань обличчя при обробці кадрів відеопотоку з веб камери в реальному часі.

2. Розроблено систему розпізнавання обличчя у відеопотоках на основі методу Віоли-Джонса й локальних бінарних шаблонів. Для виявлення обличчя у кадрах відеопотоку був використаний метод Віоли-Джонса. Класифікація

виявлених обличчя проводилася методом найближчого сусіда з використанням центральносиметричних локальних бінарних шаблонів

3. Розроблена система може використовуватися при розв'язаннях різних завдань відеоаналітики, і, у першу чергу, має безпосереднє застосування в системах контролю доступу й ідентифікації обличчя. При цьому невисокі системні вимоги уможливлюють застосування розробки на системах з низькою продуктивністю. Основними напрямками подальшого розвитку розробленого методу можна назвати поліпшення роботи класифікатора обличчя. Для цього доцільно використовувати більш досконалі алгоритми класифікації, чому використаний метод найближчого сусіда, наприклад, такі методи як Random Forests і метод опорних векторів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Video Analytics Market worth $3,971.2 Million by 2020 // Markets and Markets. URL: http://www.marketsandmarkets.com / Press Releases /i va.asp.

2. Gupta V., A Study of Various Face Detection Methods // International Journal of Advanced Research in Computer and Communication Engineering / V. Gupta, D. Sharma // Vol. 3, May 2014, №5. Pp. 6694–6697.

3. Viola P., Rapid object detection using a boosted cascade of simple features / P. Viola, M. Jones. // 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Vol. 1. 8–14 December 2001 / The Institute of Electrical and Electronics Engineers, Inc. Pp. 511–518.

4. Freund Y., A Short Introduction to Boosting // Journal of Japanese Society for Artificial Intillegence / Y. Freund, R. E. Schapire // 1999, №14(5), С. 771-780.

5. D. Roth, The SNoW Learning Architecture // Technical Report UIUCDCS-R-99-2102. UIUC Computer Science Department, 1999. - Pp. 486 – 493.

6. Nilsson M., The successive mean quantization transform / M. Nilsson, M. Dahl, I. Claesson // Proceedings of IEEE Int. conf. ICASSP 2005, Vol. 4. / The Institute of Electrical and Electronics Engineers Signal Processing Society С. 429 – 432.

7. Rowley H.A. Neural Network-Based Face Detection / H.A. Rowley, S. Baluja, T. Kanade // PAMI, January 1998. Pp. 1 – 28.

8. E. Osuna, R. Freund, F. Girosi Training support vector machines: an application to face detection // In Proceedings of Computer Vision and pattern Recognition 1997, С. 130-136.

9. Machine learning methods // Graphicon. URL: http://www.graphicon.ru/oldgr/ru/publications/text/gc2006avezh.pdf (дата звернення: 28.04.2020).

10. Rawlinson T., Principles and Methods for Face Recognition and Face Modelling / T. Rawlinson, A. Bhalerao, L. Wang // Handbook of research on computational forensics, digital crime and investigation: methods and solutions. IGI Global, C. 53-78.

11. M. Turk, A. Pentland. Eigenfaces for recognition. // Cognitive Neuroscience, 1991, №3(1), C.71–86.

12. Belhumeur P. N., Fisherfaces Recognition Using Class Specific Linear Projection / P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman // IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, July 1997, №7. - C. 711–720.

13. Ojala T., A Comparative Study of Texture Measures with Classification Based on Feature Distributions / T. Ojala, M. Pietikäinen, D. Harwood. // Pattern Recognition, Vol. 29, 1996, C. 51–59.

14. Ahonen T., Face Description with Local Binary Patterns: Application to Face Recognition / T. Ahonen, A. Hadid, // IEEE Trans. Pattern Analysis and Machine Intelligence, 1996, №28(12), Pp. 2037–2041.

15. Edwards G.J., Face Recognition Using Active Appearance Models / G.J. Edwards, T.F. Cootes, C.J. Taylor // Computer Vision — ECCV’98, Volume 1407 of the series Lecture Notes in Computer Science. Pp. 581–595.

16. http://www.biometrics.gov/Documents/FaceRec.pdf // ECE Department Carnegie Mellon University (дата звернення 18.05.2020)

17. Zhao W.Y., Image-based Face Recognition — Issues and Methods / W.Y. Zhao, R. Chellappa. // Image recognition and Classification, 2002, Pp. 375–402

18. Гонсалес Р., Цифровая обработка изображений, / Р. Гонсалес, Р. Вудс. // М.: Техносфера, 2005. – 1072 с.

19. Местецкий Л.М. Математические методы распознавания образов, М.: МГУ, ВМиК, 2002–2004. – 85 с.

20. Papageorgiou O., A general framework for object detection / O. Papageorgiou, P. Papageorgiou // International Conference on Computer Vision, 1998 / The Institute of Electrical and Electronics Engineers, Inc. C. 555–562.

21. Shapiro L. G, Computer / Vision. L. G. Shapiro, G. C. Stockman // Prentice Hall, 2001. – 608 c.

22. Heikkilä M., A texture-based method for modeling the background and detecting moving objects / M. Heikkilä, M. Pietikäinen. // IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, №28(4), C. 657-662.

23. Heikkil M., Description of Interest Regions with Center-Symmetric Local Binary Patterns / M. Heikkil, M. Pietikainen, C. Schmid // ICVGIP 2006, Pp. 58–69.

24. Броневич А. Н. Лекции по методам машинного обучения // URL: http://window.edu.ru/resource/800/73800/files/lect_Lepskiy_Bronevich_pass.pdf (дата звернення: 05.05.2020).

25. Cambrige Face Database // Cambrige university. URL: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html (дата звернення: 06.05.2020).

26. Yale Face Database B // UCSD Computer Vision. URL: http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html (дата звернення: 06.05.2020).

27. Хейлсбер А., Язык программирования C#. Классика Computers Science. 4-е изд. / А. Хейлсбер, М. Торгерсен, С. Вилтамут, П. Голд. // СПб.: Питер, 2011, 784 с.

28. About OpenCV // URL: http://opencv.org/about.html (дата звернення: 08.05.2020).

29. EmguCV // URL: http: //www.emgu.com/wiki/index.php/ Main_Page (дата звернення: 08.05.2020).

30. Удосконалення математичної моделі Віоли-Джонса для розпізнавання обличчя у відео потоці // URL: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&u

act=8&ved=2ahUKEwjZ_Lavvf7pAhWryKYKHVzXA20QFjAAegQIAxAB&url=http%3A%2F%2Fwww.tech.vernadskyjournals.in.ua%2Fjournals%2F2018%2F5_2018%2Fpart_2%2F17.pdf&usg=AOvVaw0udKGwvltfFwGTyvV3ho6f (дата звернення: 25.05.2020)

ДОДАТОК А

ЛІСТИНГ ПРОГРАМИ

```
//клас, що описує розпізнане обличчя
//підключення бібліотек using System;
using System.Windows.Forms; using System.Drawing; using System.Collections.Generic; using System.Linq; using System.Text;
using System.Threading.Tasks;

//підключення бібліотеки Emgu CV using Emgu.CV; using Emgu.CV.Structure; using Emgu.CV.Features2D; using Emgu.CV.UI;

namespace CSLBPH
{
   class Detectedface



   {
      private Faceclass _faceclass; //клас обличчя
      private Rectangle _rect; //область кадру, що містить обличчя private Densehistogram[] _histogram; //LBP гістограма обличчя private Image<Gray, Byte> _face; // зображення обличчя private Image<Gray, Byte> _lbp; //LBP зображення обличчя private double _distance; // відстань до класу

    //конструктор класу
      public Detectedface(Faceclass fc, Rectangle r, Image<Gray, Byte> f, Image<Gray, Byte> l,
   Densehistogram[] h, double d)
      {
```

```
            _faceclass = fc;
            _rect = r;
            _face = f;
            _lbp = l;
            _histogram = h;
            _distance = d;
        }

//далі – сетери й гетери
        public void Setfaceclass(Faceclass fc)
        {
            _faceclass = fc;
        }

        public void Setrect(Rectangle r)
        {
            _rect = r;
        }
        public void Setface(Image<Gray, Byte> f)
        {
            _face = f;
        }
        public void Setlbp(Image<Gray, Byte> l)
        {
            _lbp = l;
        }
        public void Sethist(Densehistogram[] h)
        {
            _histogram = h;
        }
```

```
public void Setdist(double d)
{
   _distance = d;
}
public Faceclass Getfaceclass()
{
   return _faceclass;
}

public Rectangle Getrect()
{
   return _rect;
}
public Image<Gray, Byte> Getface()
{
   return _face;
}
public Image<Gray, Byte> Getlbp()
{
   return _lbp;
}
public Densehistogram[] Gethist()
{
   return _histogram;
}
public double Getdist()
{
   return _distance;
}
```

```
    }
}
```

//клас, що описує категорію обличчя //підключення бібліотек using System; using System.Drawing; using System.Collections.Generic; using System.Linq; using System.Text; using System.Threading.Tasks;

//підключення бібліотеки Emgu CV using Emgu.CV; using Emgu.CV.Structure; using Emgu.CV.Features2D; using Emgu.CV.UI;

```
namespace CSLBPH
{
    [Serializable]    class Faceclass
    {
        private Image<Gray, Byte> _img; //зображення обличчя        private Mat[] _histogram; //LBP гістограма
        private String _name; //ім'я класу
```

//конструктор класу   public Faceclass(Image<Gray, Byte> faceimg, Mat[] facehist, String facename)

```
        {
            _img = faceimg;
            _histogram = facehist;
            _name = facename;
        }
```

//далі гетери й сетери   public Mat[] Gethist()

```
        {
            return _histogram;
        }
```

```
        public String Getname()
        {
            return _name;
        }
        public Image<Gray, Byte> Getimg()
        {
            return _img;
        }
    }
}

//клас, що описує детектор обличчя
//підключення бібліотек using System;
using System.Windows.Forms; using System.Drawing; using System.Collections.Generic; using System.Linq; using System.Text;
using System.Threading.Tasks;

//підключення бібліотеки Emgu CV using Emgu.CV; using Emgu.CV.Structure; using Emgu.CV.Features2D; using Emgu.CV.UI;

namespace CSLBPH
{
    class Facedetector
    {
        private Cascadeclassifier _haar; //каскадний класифікатор

 //конструктор класу
        public Facedetector()
        {
     //завантаження каскадів хаара для розпізнавання обличчя
```

```
            _haar = new Cascadeclassifier("haarcascade_frontalface_default.xml");
        }

    //повертає список прямокутних областей кадру з обличчями
        public Rectangle[] Getfacesrect(Image<Bgr, Byte> frame, double scalefactor, int minneighbors, int sz)
        {           try
            {
                Rectangle[] rect = _haar.Detectmultiscale(frame, scalefactor, minneighbors, new Size(sz, sz));
                Rectangle[] rect1 = new Rectangle[rect.Length];

                return rect;

            }
            catch (Argumentoutofrangeexception)
            {
                return null;
            }
        }
    }
}
```

//головний клас додатка //підключення бібліотек using System;

using System. Collections.Generic; using System. Componentmodel; using System.Data; using System.Drawing; using System.Linq; using System.Text; using System.Threading; using System. Threading.Tasks; using System.Windows.Forms; using System.IO;

using System.Runtime.Serialization.Formatters.Binary;

```
//підключення бібліотеки Emgucv using Emgu.CV; using Emgu.CV.Structure; using Emgu.CV.UI; using Zedgraph;

namespace CSLBPH
{
   public partial class Mainform : Form
   {
      private Facedetector _detector; //детектор обличчя private List<Faceclass> _faces; //класи обличчя
      private List<Recognizedface> _recognizedfaces; //розпізнані обличчя private List<Rectangle> _notrecognized; //нерозпізнані обличчя private Image<Gray, Byte> _mask; //маска значимих областей
      private Image<Gray, Byte>[] _currentfaces; //зображення для формування нового класу private int _currentfacescount; //їхня кількість private Rectangle[] _facesrect; //виявлені обличчя private bool _capturing; //прапор активності захоплення відеопотоку private Capture _capture = null; //захоплення відеопотоку private String _videopath = null; // шлях до відеофайлу private Mat _frame; //кадр відеопотоку
      private Image<Bgr, Byte> _frameimg; //кадр відеопотоку

    //конструктор класу
      public Mainform()
      {
         Initializecomponent();
         _detector = new Facedetector();
         _faces = new List<Faceclass>();
         _notrecognized = new List<Rectangle>();
         _recognizedfaces = new List<Recognizedface>();
         _currentfaces = new Image<Gray, Byte>[1000];
         _currentfacescount = 0;
```

```
            _facesrect = null;
            _mask = new Image<Gray, byte>("mask.png");
            _capturing = false;
            _capture = new Capture();
             zedgraphcontrol1.Graphpane.Title.Text = "Histogram";
zedgraphcontrol1.Graphpane.Xaxis.Title.Text = "Value";
zedgraphcontrol1.Graphpane.Yaxis.Title.Text = "Count";
        }

    //розрахунки гістограми
        public Mat Calchistogram(Image<Gray, Byte> image)
        {
            int dim = 4; // розбівка на блоки
            Mat result = new Mat();
             int width = image.Width / dim;
            int height = image.Height / dim;

        //цикл по всіх блоках
            for (int i = 0; i < dim; i++)
            {
                for (int j = 0; j < dim; j++)
                {
                    image.ROI = new Rectangle(i * width, j * height, width, height);
Densehistogram hist = new Densehistogram(16, new Rangef(0, 15));
                    Image<Gray, Byte>[] inp = new Image<Gray, Byte>[1];
inp[0] = image;

                                    //розрахунки для поточного блоку
                    hist.Calculate(inp, true, null);
                    Cvinvoke.Normalize(hist, hist);
```

```
                                    //додавання до підсумкової гістограмі
                    result.Pushback(hist);
                }
            }

            image.ROI = Rectangle.Empty;            return result;
        }

    //розпізнавання обличчя
        private void Recognizeface(Rectangle rect, Image<Gray, Byte> face, Image<Gray, Byte> lbp,
    Mat histogram, decimal threshold)
        {
            double mindist = double.Maxvalue;//мінімальна відстань до класу Faceclass answer = null; // клас-результат

                //цикл по всіх класах обличчя            foreach (Faceclass f in _faces) {  //цикл по всім гістограмам класу              foreach (Mat h in f.Gethist())
                {
                    double dist = Cvinvoke.Comparehist(histogram, h, Emgu.CV.Cvenum.Histogramcompmethod.Chisqr); //розсіє відстані

                    if (dist <= mindist) //зрівняти відстань із мінімальним
                    {
                        mindist = dist; //обновити мінімальну відстань answer = f; //прийняти результатом даний клас обличчя
                    }
                }
            }
```

```
if (mindist < (double)threshold) //якщо відстань менше граничного обновити список розпізнаних обличь {
bool contains = false;
Cvinvoke.Equalizehist(lbp, lbp);

foreach (Recognizedface element in _recognizedfaces)
{
if ((element.Getfaceclass().Getname() == answer.Getname()) && (mindist < element.Getdist())) {
_notrecognized.Add(element.Getrect());
element.Setface(face); element.Setlbp(lbp); element.Setrect(rect); element.Sethist(histogram); element.Setdist(mindist); contains = true;
}
else if ((element.Getfaceclass().Getname() == answer.Getname()) && (mindist
>= element.Getdist())) //інакше додати обличчя в список нерозпізнаних облич
{
contains = true; _notrecognized.Add(rect);
}
}

if (!contains)
{
_recognizedfaces.Add(new Recognizedface(answer, rect, face, lbp, histogram, mindist)); } } else
{
_notrecognized.Add(rect);
```

```
            }
        }

    //обробка списку облич
        private void Processfaces()
        {
            foreach (Rectangle rect in _facesrect) // для кожного обличчя в списку виявлених
            {
                _frameimg.ROI = rect;
                Image<Gray, Byte> face = _frameimg.Convert<Gray, Byte>();
                _frameimg.ROI = Rectangle.Empty;
                face = face.Resize(128, 128, Emgu.CV.Cvenum.Inter.Linear); //масштабування
                Cvinvoke.Gaussianblur(face, face, new Size((int)blurupdown.Value * 2 - 1,
    (int)blurupdown.Value * 2 - 1), 0); // розмиття по Гаусу

                Image<Gray, Byte> lbpnomask = Lbp-transformer.Cstransform(face); //LBP перетворення
                Image<Gray, Byte> lbp = new Image<Gray, byte>(128, 128, new Gray(0));
                Cvinvoke.cvcopy(lbpnomask, lbp, _mask);
                Mat histogram = Calchistogram(lbp); //розрахунки гістограмми
                Recognizeface(rect, face, lbp, histogram, thresholdupdown.Value); //розпізнавання
            }
        }

    //обробка кадру
        private void Processframe(object sender, Eventargs arg)
```

```
            {
                _recognizedfaces = new List<Recognizedface>(); //обнуління списків обличчя
                _notrecognized = new List<Rectangle>();
                _frameimg = new Image<Bgr, Byte>(_capture.Width, _capture.Height);
                _frame = new Mat();
                _capture.Retrieve(_frame, 0); //одержання кадра відеопотоку
                _frameimg = _frame.Toimage<Bgr, Byte>();

                _facesrect = _detector.Getfacesrect(_frameimg, (double)scalefactorupdown.Value,
        (int)neighborsupdown.Value, (int)minsizeupdown.Value); //виявлення обличчя

                if (_facesrect.Length == 1)
                {
                    addfacebutton.Invoke(new Action(() => { addfacebutton.Enabled = true; }));
                } else {
                    addfacebutton.Invoke(new Action(() => { addfacebutton.Enabled = false; }));  }

                Processfaces(); //обробка списку виявлених обличчя
                Drawdetected(); //висновок даних
            }

        //очищення даних з форми
            private void Cleardata()
            {
```

```
            zedgraphcontrol1.Graphpane.Curvelist.Clear();
zedgraphcontrol1.Axischange();                zedgraphcontrol1.Invalidate();
capfaceimagebox.Image = null;          lbpimagebox.Image = null;

            distlabel.Invoke(new Action(() =>
            {
               distlabel.Text = " ";
            }));

         }

      //висновок даних на форму
         private void Drawdetected()
         {
            bool contains = false;

            foreach (Recognizedface face in _recognizedfaces)//цикл повсем
розпізнаним обличчям          {
               _frameimg.Draw(face.Getrect(), new Bgr(0, 255, 0), 3);//виділити
зеленим у кадрові
               _frameimg.Draw(face.Getfaceclass().Getname(),
face.Getrect().Location,
      Emgu.CV.Cvenum.Fontface.Hersheycomplex, 1, new Bgr(0, 255, 0));
//написати в кадрові ім'я класу
                        //висновок даних на форму
               faceclasseslistbox.Invoke(new Action(() =>
               {
                  if                (face.Getfaceclass().Getname()              ==
(string)faceclasseslistbox.Selecteditem)
                  {
```

```
                    capfaceimagebox.Image = face.Getface();
lbpimagebox.Image = face.Getlbp();                    Drawhistogram(face.Gethist());
distlabel.Text = face.Getdist().Tostring();                    contains = true;
                }
            }));
        }

        if (!contains) Cleardata();

        foreach (Rectangle rect in _notrecognized) //виділити всі нерозпізнані обличчя червоним        {
            _frameimg.Draw(rect, new Bgr(0, 0, 255), 3);
        }

        _frameimg = _frameimg.Resize(capturebox.Width, (int)(_frame.Height * (capturebox.Width / (double)_frame.Width)), Emgu.CV.Cvenum.Inter.Linear);
capturebox.Image = _frameimg;
    }

    //функція гістограми на формі        private void Drawhistogram(Mat histogram)
    {
        Graphpane pane = zedgraphcontrol1.Graphpane;
pane.Curvelist.Clear();        int dim = histogram.Height;

        Image <Gray, Double> histimg = histogram.Toimage<Gray, Double>();
        double[] hist = new double[dim];                double[] xvalues = new double[dim];

        for (int i = 0; i < dim; i++)
```

```
            {
                xvalues[i] = i;
                hist[i] = histimg.Data[i, 0, 0];
            }

            Baritem bar1 = pane.Addbar("", xvalues, hist, Color.Deepskyblue); zedgraphcontrol1.Axischange();          zedgraphcontrol1.Invalidate();
        }

    //відкриття класу обличчя
        private void openfaceclass(object sender, Canceleventargs e)
        {
            foreach (string name in openfiledialog1.Filenames)
            {
                if (File.Exists(name))
                {
                    Stream Testfilestream = File.Openread(name);
                    Binaryformatter deserializer = new Binaryformatter();
                    Faceclass                              face                              = (Faceclass)deserializer.Deserialize(Testfilestream);
                    Testfilestream.Close();                _faces.Add(face);
                    faceclasseslistbox.Items.Add(face.Getname());
                }
            }
        }

         //додавання зображення обличчя в список для формування класу private void addfacebutton_Click(object sender, Eventargs e)
        {
            if ((_currentfacescount < 1000) && (_facesrect.Length != 0))
```

```
            {
                for (int qwe = 0; qwe < 1000; qwe++)            {

                    Image<Gray, Byte> fr = _frame.Toimage<Gray, Byte>();  fr.ROI = _facesrect [0];

                    _currentfaces [_currentfacescount] = fr.Resize(128, 128, Emgu.CV.Cvenum.Inter.Linear);
                    Cvinvoke.Gaussianblur(_currentfaces[_currentfacescount], _currentfaces[_currentfacescount],
                        new Size((int)blurupdown. Value * 2 - 1, (int)blurupdown.Value * 2 - 1), 0);
                    imagebox4.Image = _currentfaces[_currentfacescount];
                    _currentfacescount++;
                    countlabel. Text = _currentfacescount.Tostring(); addclassbutton.Enabled = true;                removefacebutton.Enabled = true;
                }
            }
        }

    //додавання нового класу обличчя
        private void addclassbutton_Click(object sender, Eventargs e)
        {
      //ініціалізація
            Image<Gray, Byte>[] img = new Image<Gray, Byte>[_currentfacescount];
            Image<Gray, Byte> face = _currentfaces[0].Resize(128, 128, Emgu.CV.Cvenum.Inter.Linear);
            Mat[] hist = new Mat[_currentfacescount];
```

```
//обробка облич списку формування нового класу й розрахунки гістограм
for (int i = 0; i < _currentfacescount; i++)
{
Image<Gray, Byte> lbpnomask = Lbptransformer.Cstransform(_currentfaces[i]); img[i] = new Image<Gray, byte>(128, 128, new Gray(0));
Cvinvoke.cvcopy(lbpnomask, img[i], _mask); hist[i] = Calchistogram(img[i]);
}

_faces.Add(new Faceclass(face, hist, classnametextbox.Text));

faceclasseslistbox.Items.Add(classnametextbox.Text);

_currentfaces = new Image<Gray, Byte>[1000];
_currentfacescount = 0; countlabel.Text = "0";
addclassbutton.Enabled = false; removefacebutton.Enabled = false; imagebox4.Image = null;
}

//зміна обраного класу обличчя
private void faceclasseslistbox_Selectedindexchanged(object sender, Eventargs e)
{
foreach (Faceclass face in _faces)
{
if (face.Getname() == (string)faceclasseslistbox.Selecteditem)
{
classimageimagebox.Image = face.Getimg();
}
```

```
            }

            if (faceclasseslistbox.Selecteditem != null)
            {
                saveclassbutton.Enabled = true;          removeclassbutton.Enabled
= true;
            }          else          {
                saveclassbutton.Enabled = false;          removeclassbutton.Enabled
= false;
            }
        }

    //видалення класу обличчя зі списку
        private void removefacebutton_Click(object sender, Eventargs e)
        {
            _currentfacescount--;
            countlabel.Text  =  _currentfacescount.Tostring();                    if
(_currentfacescount != 0)
            {
                imagebox4.Image = _currentfaces[_currentfacescount - 1];
            } else          {
                removefacebutton.Enabled = false;          addclassbutton.Enabled
= false;            imagebox4.Image = null;
            }
        }

    //запуск обробки відеопотоку
        private void videobutton_Click(object sender, Eventargs e)
        {
            if (_capturing)
```

```
        {
            startbutton.Text = "Start";
            _capturing = false;
            _capture.Stop();
        }           else          {              try              {
                startbutton.Text = "Stop";
                _capturing = true;

                if (radiobutton1.Checked)
                {
                    _capture.Dispose();
                    _capture = new Capture();
                    _capture.Imagegrabbed += Processframe;
                    _capture.Start();
                }                   else
                {
                    _capture.Dispose();
                    _capture = new Capture(_videopath);
                    _capture.Imagegrabbed += Processframe;
                    _capture.Start();
                }               }
            catch (Nullreferenceexception excpt)
            {
                Messagebox.Show(excpt.Message);
            }
        }

    }
```

//збереження класу обличчя у файл

```
        private void savefaceclass(object sender, Canceleventargs e)
        {
            Faceclass tosave = null;

            foreach (Faceclass face in _faces)
            {
                if (face.Getname() == faceclasseslistbox.Selecteditem.Tostring())
                {
                    tosave = face;
                }
            }

            Stream filestream = File.Create(savefiledialog1.Filename);
Binaryformatter serializer = new Binaryformatter();
serializer.Serialize(filestream, tosave); filestream.Close();
        }

    //далі оброблювачі натискань кнопок форми
        private void saveclassbutton_Click(object sender, Eventargs e)
        {
            savefiledialog1.Showdialog();
        }
        private void openclassbutton_Click(object sender, Eventargs e)
        {
            openfiledialog1.Showdialog();
        }
        private void removeclassbutton_Click(object sender, Eventargs e)
        {
```

```
                _faces.Removeall(f => f.Getname() == (string)faceclasseslistbox.Selecteditem);
faceclasseslistbox.Items.Remove(faceclasseslistbox.Selecteditem);        }
            private void openvideobutton_Click(object sender, Eventargs e)
            {
               openfiledialog2.Showdialog();
            }
            private void openfiledialog2_Fileok(object sender, Canceleventargs e)
            {
               _videopath = openfiledialog2.Filename;            videopathlabel.Text = _videopath;
            }

         }
      }
```

ДОДАТОК Б

ABSTRACT

The work is devoted to the development of face recognition system in video streams in real time. Detection system is made by using the method of Viola Jones. The classification is made by using the nearest neighbor method and histograms of centrally symmetric local binary patterns as signs of classification. In addition, within the work the research of the effectiveness of various modifications of the local binary patterns was made. This research showed the high efficiency of centrally symmetric local binary patterns applied to face recognition task in real time.